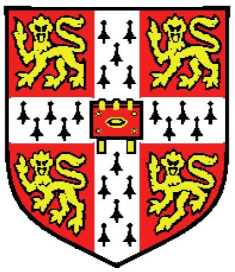
Penalised regression for high-dimensional data: an empirical investigation and improvements via ensemble learning



Fan Wang

Wolfson College

University of Cambridge

This dissertation is submitted for the degree of

Doctor of Philosophy

08/18

Declaration

I declare that the contents presented in this dissertation are the results of my own work except where specified otherwise. I declare that this thesis was composed by myself, and no part of this thesis has been submitted for a degree or other qualification at any university.

The work in Chapter 2 together with parts of Chater 3 have been submitted for publication. The manuscript is also available as an arXiv preprint [Wang et al., 2018]. The work in Chapter 4 will shortly be submitted for publication.

Fan Wang

Acknowledgements

First of all, I would like to thank my supervisors Sylvia Richardson and Steven Hill, for their unwavering support during my PhD studies. Their sharp intuition and deep insights over the past years have been invaluable and have enabled me to finish the thesis much more easily. Their constant patience and encouragement have been of great help, not only for my research, but also for my future career.

I would like to express my appreciation to all the support I received from colleagues and advisers. Particularly, I am deeply grateful to Sach Mukherjee, Gwenael Leday, Paul Kirk and Paul Newcombe for their great advice on improving my work. I wish to thank all members of the MRC Biostatistics Unit for the pleasurable working atmosphere.

My gratitude also goes to all my close friends for their unconditional support in my life. They help me come out of difficult times, and their sharing and listening have meant a lot to me. Particularly, I would like to thank Qiyu Yuan, Alex Yuan, Peicheng Xu, Tien Vo, Jason Xu and Mary Caplinger for always being there for me.

Finally, I would like to thank my family. Without their love and care, I would not have been able to come this far.

Summary

In a wide range of applications, datasets are generated for which the number of variables p exceeds the sample size n. Penalised likelihood methods are widely used to tackle regression problems in these high-dimensional settings. In this thesis, we carry out an extensive empirical comparison of the performance of popular penalised regression methods in high-dimensional settings and propose new methodology that uses ensemble learning to enhance the performance of these methods.

The relative efficacy of different penalised regression methods in finite-sample settings remains incompletely understood. Through a large-scale simulation study, consisting of more than 1,800 data-generating scenarios, we systematically consider the influence of various factors (for example, sample size and sparsity) on method performance. We focus on three related goals — prediction, variable selection and variable ranking — and consider six widely used methods. The results are supported by a semi-synthetic data example. Our empirical results complement existing theory and provide a resource to compare performance across a range of settings and metrics.

We then propose a new ensemble learning approach for improving the performance of penalised regression methods, called STructural RANDomised Selection (STRANDS). The approach, that builds and improves upon the Random Lasso method, consists of two steps. In both steps, we reduce dimensionality by repeated subsampling of variables. We apply a penalised regression method to each subsampled dataset and average the results. In the first step, subsampling is informed by

variable correlation structure, and in the second step, by variable importance measures from the first step. STRANDS can be used with any sparse penalised regression approach as the "base learner". In simulations, we show that STRANDS typically improves upon its base learner, and demonstrate that taking account of the correlation structure in the first step can help to improve the efficiency with which the model space may be explored.

We propose another ensemble learning method to improve the prediction performance of Ridge Regression in sparse settings. Specifically, we combine Bayesian Ridge Regression with a probabilistic forward selection procedure, where inclusion of a variable at each stage is probabilistically determined by a Bayes factor. We compare the prediction performance of the proposed method to penalised regression methods using simulated data.

Contents

Abbreviations

AdaLasso	adaptive Lasso
AIC	Akaike information criterion
AUC	area under the ROC curve
bagging	bootstrap aggregation
BF	Bayes factor
BIC	Bayesian information criterion
Bolasso	Bootstrap-enhanced Lasso
ENet	elastic net
fMRI	functional magnetic resonance imaging
FN	false negative
FP	false positive
FPR	false positive rate
GWAS	genome-wide association study
HENet	heavy elastic net
LARS	least angle regression
Lasso	least absolute shrinkage and selection operator
LENet	light elastic net
MLE	maximum likelihood estimate
NC	no clustering
NIG	normal inverse gamma
pAUC	partial area under the ROC curve
PPV	positive predictive value
RC	random clustering
RFS	adaptive Ridge forward selection
RLasso	random Lasso
ROC	receiver operating characteristic
RSS	residual sum of squares
SCAD	smoothly clipped absolute deviation
SIS	sure independence screening
SNR	signal-to-noise ratio
STRANDS	structural randomised selection
TN	true negative
TP	true positive
TPR	true positive rate

Chapter 1

Introduction

1.1 Statistical challenges in modern applications of linear regression

Thanks to the rapid advance of technologies for data acquisition, the complexity of data has increased in many research areas, including finance, technology and science. As a result, the number of variables being analysed can be huge, and can easily surpass the number of available samples. These data pose challenges to traditional statistical approaches, yet they offer unprecedented opportunities for us to understand the underlying patterns behind phenomena.

With the advent of high-throughput technology, we are now able to measure the abundance of large numbers of gene and protein expression profiles, as well as sequence whole genomes. The enormous amount of information contained in data generated by these technologies enables us to investigate biological processes at different stages of protein synthesis. These data significantly facilitate in-depth studies of biological processes that govern functions performed by cells. For example, genome-wide association studies [GWAS, Rudan et al., 2016] link diseases with genetic variants (often single nucleotide polymorphisms, or SNPs); differential expression analysis [Costa-Silva et al., 2017] aims to discover significant differences in gene expression (mRNA or cDNA) of different cells, and find tissue specific patterns of expression; feature extraction and functional modelling in proteomics shed light upon mechanism of molecular regulatory processes by detecting poten-

tial protein biomarkers of disease, which will eventually lead to clinical application [Morris, 2012]; and epigenetic data analysis allows for the discovery of connections between quantifiable epigenetic marks and traits of interest [Rakyan et al., 2011]. Treatment based on biological information specific to an individual person has the potential of high efficiency and effectiveness, which is receiving more and more attention in personalised medicine research. High-throughput data has its unique challenges, including high dimension, high levels of noise, complex structure such as multicollinearity, systematic bias, and heterogeneity arising from aggregation of datasets or from subpopulations which may be unknown. They give rise to the need of novel statistical methodology and computational approaches.

Study of high-throughput data aims to explain observed variation in a phenotype of interest with molecular variants and environmental factors. Regression is a powerful tool for modelling the relationship between responses and potential features, and is widely applied in biomedical studies. Linear regression is arguably the simplest technique in regression. Besides computational efficiency, linear regression models have good interpretability because they establish explicit relationship between the features and the outcome.

1.2 Linear regression model

Assume the data consists of n samples and p variables. $\mathbf{Y} \in \mathbb{R}^n$ is a $n \times 1$ response vector, $\mathbf{X}$ is a $n \times p$ design matrix of p variables $(\boldsymbol{x_1}, \ldots, \boldsymbol{x_p})$, $\boldsymbol{x_j} \in \mathbb{R}^n$ for $j = 1 \ldots p$. The Gaussian linear regression model takes the form

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{1.1}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ is a $p \times 1$ vector of coefficients, and $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}_n)$ is a $n \times 1$ vector of error terms, $\mathbf{I}_n$ being the identity matrix (denoted as $\mathbf{I}$ hereafter if no confusion can occur). Throughout the thesis we assume that each column of design matrix $\mathbf{X}$ has been standardised to have mean zero and variance one, and $\mathbf{Y}$ is mean-centred. This results in the intercept term being zero and it is therefore omitted from the model (1.1).

1.2.1 Maximum likelihood estimate

The likelihood function for $\boldsymbol{\beta}$ is given by

$$l(\boldsymbol{\beta}|\mathbf{Y}, \mathbf{X}, \sigma^2) = c \cdot \exp(-\frac{1}{2\sigma^2}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^{\mathrm{T}}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})), \tag{1.2}$$

where c is a constant independent from $\boldsymbol{\beta}$. Maximising the likelihood function for $\boldsymbol{\beta}$ by differentiating $l(\boldsymbol{\beta}|\mathbf{Y}, \mathbf{X}, \sigma^2)$ with respect to $\boldsymbol{\beta}$ and setting the derivative to zero, it can be seen that the maximum likelihood estimate (MLE) is obtained by solving $(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})\hat{\boldsymbol{\beta}} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}$. If $\mathbf{X}^T\mathbf{X}$ is of full rank, this has the closed-form solution

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}. \tag{1.3}$$

The case where $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is not of full rank is discussed below. Note that, under the assumption that $\boldsymbol{\epsilon}$ follows multivariate normal distribution, the maximum likelihood estimate is the same as the least squares estimate, which is obtained by minimising the residual sum of squares (RSS)

$$RSS(\boldsymbol{\beta}) = ||\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}||_2^2, \tag{1.4}$$

where $||.||_2$ is the l_2 norm.

The MLE is an unbiased estimator for $\boldsymbol{\beta}$, and according to the Gauss−Markov theorem, it has the smallest variance in all linear unbiased estimators. Linear models have easily interpretable results, i.e., for one unit increase of a numerical variable, while holding all other variables fixed, the expected change in the outcome is the corresponding coefficient estimate. The coefficient estimates therefore represent the strength of association between response and each variable. They can be used to predict a previously unseen response given some new data $\boldsymbol{X_{new}}$, or conduct hypothesis tests to determine the statistical significance of each variable.

1.2.2 Regularisation, sparsity and high-dimensionality

In the case where $p > n$ or variables are linearly dependent, $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is singular and the MLE is not uniquely defined. Also matrix inversion will be numerically unstable when variables are highly correlated (i.e., close to linearly dependent). The

MLE can have high variance, and can be undesirable with respect to prediction accuracy and interpretation in these settings. Regularisation can be applied in these cases, which introduces additional information to solve ill-posed problems, and also avoid over-fitting. It reduces the variance of estimates by sacrificing unbiasedness, and improve the prediction accuracy. The bias-variance tradeoff is typical in regression problems; a model cannot simultaneously perfectly fit the training data, and generalise well to previously unseen data. Fewer variables in a model typically result in smaller variance of coefficient estimates. Regularisation typically specifies constraints on $\boldsymbol{\beta}$ when optimising the target function, and constraints with certain properties can lead to sparse representation.

At first glance, when $p > n$, it seems to be impossible to retrieve $\boldsymbol{\beta}$ from data, which is analogous to solving an underdetermined equation set where number of unknowns is larger than number of equations. However, with a sparsity assumption that only a few entries in $\boldsymbol{\beta}$ are non-zero, it is feasible to estimate the coefficients. In biological studies as well as other high-dimensional settings, it is usually reasonable to assume that only a small number of variables are actually associated with the outcome. Even in studies where many variables are expected to be responsible for the outcome (e.g. in genetics/genomics where there are a few strong signals and many weak ones), a standard practice is to focus on a small number of strong signals, as detecting all the signals is usually not feasible. Sparse representation not only makes regression results interpretable, but can also make the predictive model more accurate.

1.2.2.1 Sparse linear regression model

In the sparse linear regression model we assume that only a subset of the p variables have non-zero coefficients, and we refer to these variables as relevant variables or signals. Variables with zero coefficients are referred to as irrelevant variables or non-signals. We define a model to be a specific subset of variables, denoted by $\mathbf{J}=\{j_1, ...j_k\}, (1 \leq j_1 < ... < j_k \leq p)$, where each element in $\mathbf{J}$ corresponds to a variable, and $\boldsymbol{\gamma} = (\gamma_1 \dots \gamma_p)$, where $\gamma_j = 1$ if $\beta_j \neq 0$ and $\gamma_j = 0$ if $\beta_j = 0$. Let $s_0 = \sum_{j=1}^{p} \gamma_j$ denote the sparsity, $\mathbf{X_J}$ denote the design matrix composed with columns of $\mathbf{X}$ that correspond to model $\mathbf{J}$, and $\boldsymbol{\beta_J} = (\beta_{j_1}, \dots \beta_{j_k})$. Define

$\boldsymbol{\omega} = \{j : \gamma_j = 1, 1 \leq j \leq p\}$, and the true linear model can then be expressed as $\mathbf{Y} = \boldsymbol{X}_{\boldsymbol{\omega}}\boldsymbol{\beta}_{\boldsymbol{\omega}} + \boldsymbol{\epsilon}$.

There are several types of correlation that can be considered. e.g. correlation among signals, correlation among non-signals, correlation between one signal and all other variables, or correlation among all variables. In the rest of thesis we will be explicit about types of correlation when referring to, and whenever we refer to "correlated variables", we mean correlated variables in general, irrespective of whether signal or non-signal.

1.2.2.2 Goals of high-dimensional sparse linear regression

Common goals of sparse linear regression analysis include estimating coefficients to assess influence of individual variables, recovering sparsity patterns $\boldsymbol{\gamma}$ (i.e., variable selection), ranking variables and predicting previously unseen responses. These goals are related but not identical. For prediction, we are interested in using $\hat{\boldsymbol{\beta}}$ to predict response for a new sample. The specific variables that are included in the model may not be of direct importance and any set of variables that provides the same prediction risk may be acceptable. In contrast, for variable selection it is identifying the true set of relevant variables that is the goal, while we treat variable ranking as a third goal due to the fact that in many applications, practitioners are interested in receiving guidance for follow-up studies or data acquisition. Then, highlighting relevant variables in a suitable order is particularly important.

1.3 Subset selection

In settings where only a small proportion of variables are relevant, or strong multicollinearity exists, sparse models can help to reduce variance of estimates, and improve prediction accuracy and model interpretability, as discussed in Section 1.2.2. Subset selection achieves sparsity by evaluating different subsets of variables for their suitabilities, and finding the "best subset", where "best" is characterised statistically.

1.3.1 Best subset selection

Best subset selection is a straightforward approach that finds the subset of size k giving the smallest RSS for each $k = 0, \ldots, p$. The optimal choice of k should then be determined based on the tradeoff between bias and variance, also taking into account the assumed sparsity level of the underlying model. Since for p variables the number of candidate models is 2^p, best subset selection quickly becomes infeasible for large p.

1.3.2 Forward and backward stepwise selection

One way to solve the computation issue of best subset selection is to find a path through some candidate models. Forward stepwise selection, or forward selection for short, starts with the intercept term, and sequentially adds in the variable that has the most positive impact on model fit. In contrast, backward stepwise selection, or backward selection for short, starts with the full model, and sequentially removes the variable with most negative impact on model fit. Both forward and backward stepwise selection are greedy algorithms in that they produce a sequence of nested models, and optimise the model fit at each stage.

Common criteria for assessing relative model fit include Akaike information criterion (AIC) and Bayesian information criterion (BIC). In the linear regression model (1.1), AIC and BIC of a model are defined as

$$\begin{aligned} \text{AIC} &= 2k - n\log(\hat{l}) \\ \text{BIC} &= k\log(n) - n\log(\hat{l}), \end{aligned} \tag{1.5}$$

where k is the degree of freedom of the model, and $\hat{l}$ is maximum value of the likelihood function, optimised over $\boldsymbol{\beta}$. AIC and BIC penalise the number of variables in the model in order to avoid overfitting. When using AIC or BIC as model fit criteria, at each stage of forward/backward stepwise selection, an addition/deletion of a variable is performed to minimise AIC or BIC score. The procedure stops when AIC/BIC cannot further decrease.

1.3.3 Forward stagewise regression

Forward stagewise regression has a similar spirit as forward stepwise selection, but is less greedy. Specifically, let $\hat{\boldsymbol{\beta}}$ be the coefficient estimates updated at each step, and $\hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}}$ be corresponding estimate of $\mathbf{Y}$. Initialise $\hat{\boldsymbol{\mu}} = \mathbf{0}$, and regression function is built up iteratively. At each step, calculate the vector of association

$$\hat{\boldsymbol{c}} = \boldsymbol{X}^{\mathrm{T}}(\mathbf{Y} - \hat{\boldsymbol{\mu}}), \tag{1.6}$$

where $\hat{c}_j$ is proportional to the correlation between $\boldsymbol{x_j}$ and the current residual $\mathbf{Y} - \hat{\boldsymbol{\mu}}$. Both forward selection and forward stagewise regression then proceed in the direction with greatest current correlation:

$$\begin{aligned} j &= \operatorname{argmax}|\hat{c}_j| \\ \hat{\boldsymbol{\mu}} &\leftarrow \hat{\boldsymbol{\mu}} + \eta \cdot \operatorname{sign}(\hat{c}_j) \cdot \boldsymbol{x_j}, \end{aligned} \tag{1.7}$$

where η is a constant. Forward selection takes $\eta = |\hat{c}_j|$, fully adding the variable into the model, which can be greedy in the sense that the step could be too large. Variables correlated with those already selected to be in the model are likely to never enter the model, or only enter with small coefficients. Forward stagewise regression is less greedy by taking η to be small. The differences of forward selection and forward stagewise regression for $p = 2$ are illustrated in Figure 1.1. If the step size η is chosen to be very small in stagewise regression, the procedure can be computationally intensive.

1.3.4 Least angle regression

Least Angle Regression (LARS) [Efron et al., 2004] is a principled and computationally efficient version of stagewise regression. Similar to stagewise regression, LARS builds $\hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}}$ at each step, and initialises $\hat{\boldsymbol{\beta}} = (\beta_1, \beta_2, \ldots, \beta_p) = \mathbf{0}$. The procedure starts by finding the variable $\boldsymbol{x_{j_1}}$ that is most correlated with $\mathbf{Y}$, and $\hat{\boldsymbol{\mu}}$ proceeds in the direction of this variable until another variable $\boldsymbol{x_{j_2}}$ has the same amount of correlation with residual $\mathbf{r} = \mathbf{Y} - \hat{\boldsymbol{\mu}}$ as $\boldsymbol{x_{j_1}}$. LARS then proceeds in the direction that is equiangular between $\boldsymbol{x_{j_1}}$ and $\boldsymbol{x_{j_2}}$, until another variable $\boldsymbol{x_{j_3}}$ has the same amount of correlation with residual $\mathbf{r} = \mathbf{Y} - \hat{\boldsymbol{\mu}}$ as $\boldsymbol{x_{j_1}}$ and $\boldsymbol{x_{j_2}}$. LARS

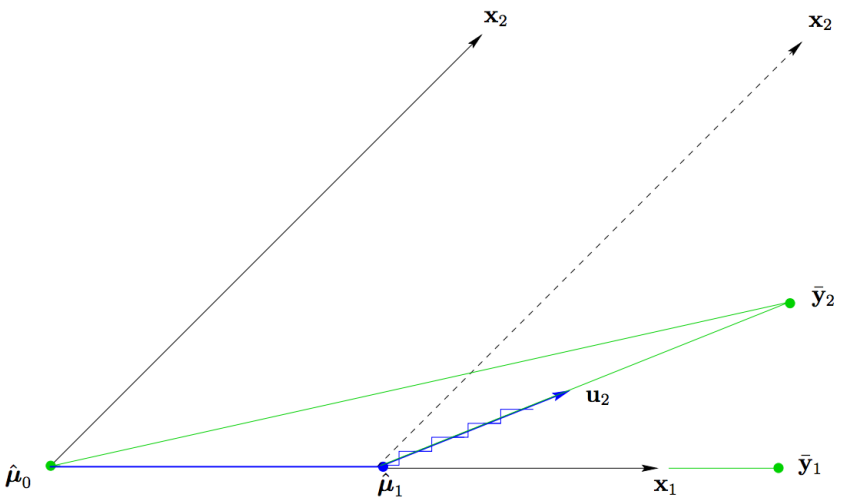


Figure 1.1: Illustration of forward selection, stagewise regression and LARS for $p = 2$. $\bar{\mathbf{y}}_2$ is the projection of $\mathbf{Y}$ on space spanned by $\boldsymbol{x_1}$, $\boldsymbol{x_2}$, and $\hat{\boldsymbol{\mu}}_\mathbf{0}$, $\hat{\boldsymbol{\mu}}_\mathbf{1}$, $\hat{\boldsymbol{\mu}}_\mathbf{2}$ denote the estimated response at steps 0,1 and 2. The initial response estimate is $\hat{\boldsymbol{\mu}}_\mathbf{0} = \mathbf{0}$, and the residual (i.e., $\bar{\mathbf{y}}_2$) has higher correlation with $\mathbf{x_1}$, so the LARS algorithm proceeds in direction of $\mathbf{x_1}$ until $\boldsymbol{\mu} = \hat{\boldsymbol{\mu}}_\mathbf{1}$, which is the point where the residuals $\bar{\mathbf{y}}_2 - \boldsymbol{\mu}$ have the same correlation with $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$, and then the algorithm proceeds along the bisector direction (indicated by $\boldsymbol{u_2}$) until $\hat{\boldsymbol{\mu}}_2 = \bar{\mathbf{y}}_2$. In contrast, forward stagewise regression takes small step size such that its path has staircase shape (purple line), and forward stepwise regression proceeds in $\boldsymbol{x_1}$ direction further than LARS until $\bar{\mathbf{y}}_1$, the projection of $\mathbf{Y}$ on to $\boldsymbol{x_1}$. [Figure reproduced from Efron et al., 2004]

then proceeds in the direction that is equiangular between $\boldsymbol{x_{j_1}}$, $\boldsymbol{x_{j_2}}$, $\boldsymbol{x_{j_3}}$. This process continues until all variables are included.

LARS is computationally efficient since it can produce piecewise linear solution path with p steps. It can also be adapted to provide the solution path for some other methods such as Lasso (see Section 1.4.2.2), which significantly accelerates the computation. Finally, the LARS estimator tends to assign highly correlated features with similar coefficient estimates, which is a desirable property. An illustration of LARS algorithm for $p = 2$ is shown in Figure 1.1.

1.4 Penalised regression

Conventional variable selection approaches may not be suitable for high-dimensional data. For instance, best subset selection fails to cope with large number of variables, since it is extremely variable due to the nature of discreteness, and computation can be prohibitive, because the number of potential sub-models grows exponentially with dimensionality. Greedy algorithms such as forward and backward selection can be unstable with high variance due to their discontinuous property, i.e., a small change in the data can lead to completely different estimates. Penalised likelihood regression methods achieve variable selection in a continuous and computationally efficient way, and suffer less variability. Below, we first provide an introduction to penalised likelihood regression model in general and then describe some specific methods.

1.4.1 Introduction to penalised regression

Penalised regression methods augment the loss function with a penalty term that encodes a structural assumption such as sparsity. Specifically, in penalised likelihood regression, instead of maximizing the likelihood function $l(\boldsymbol{\beta}|\mathbf{Y},\mathbf{X})$, one maximises the objective function

$$M(\boldsymbol{\beta}) = l(\boldsymbol{\beta}|\mathbf{Y},\mathbf{X}) - P_\lambda(\Theta), \tag{1.8}$$

where $P_\lambda(.)$ is a penalty function that penalises complex models, and $\lambda > 0$ controls the degree of penalisation. In linear regression model, it is equivalent to minimising the objective function using RSS as the loss function:

$$M'(\boldsymbol{\beta}) = ||\mathbf{Y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + P_\lambda(\boldsymbol{\beta}). \tag{1.9}$$

Throughout the thesis, we use (1.9) as the objective function for penalised linear regression. $P_\lambda(\boldsymbol{\beta})$ shrinks some coefficient estimates towards zero, and when $P_\lambda(\boldsymbol{\beta})$ satisfies certain properties, some of the coefficients are shrunk to exactly zero, leading to a sparse solution.

Unlike the MLE, the solution for minimizing (1.9) depends on the value of

λ. The choice of λ can be data-driven, such as minimising information criteria or through cross-validation. Specifically, when using information criteria, λ is chosen such that optimal information criteria score is achieved. Common criteria include AIC, BIC and for ultra-high dimensional data, extended BIC (EBIC) [Chen and Chen, 2008]. Cross-validation is a technique to evaluate how well model estimates can be generalised to an independent unseen data set. Specifically for linear regression, in a k-fold cross-validation, the original samples are randomly partitioned into k subsets of the same size. One subset is treated as validation data, while the rest of the subsets are treated as training data. The model is fitted on the training data while prediction accuracy is evaluated using the validation data to give a score. The process is repeated k times (folds), such that each subset is used as validation data exactly once. Scores are then averaged across folds to produce an estimate for out-of-sample prediction accuracy. λ that minimises the cross-validation prediction error is considered optimal for the model to generalise to an independent dataset.

1.4.2 Penalised regression methods

In this section we introduce several popular penalised regression methods for high-dimensional data analysis. Below, we will use the following definitions about variable selection consistency, estimation consistency and prediction consistency. Let $\hat{\boldsymbol{\beta}}$ be an estimate of $\boldsymbol{\beta}$.

Definition $\hat{\boldsymbol{\beta}}$ is a consistent estimate if

$$\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} \to \mathbf{0}, \text{as } n \to \infty, \tag{1.10}$$

Definition $\hat{\boldsymbol{\beta}}$ is selection consistent if

$$P(\{j : \hat{\beta}_j \neq 0\} = \{j : \beta_j \neq 0\}) \to 1, \text{as } n \to \infty \tag{1.11}$$

Definition $\hat{\boldsymbol{\beta}}$ is prediction consistent if

$$||\mathbf{X}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})||_2^2/n \to 0, \text{as } n \to \infty \tag{1.12}$$

In order to illustrate the main properties of these methods, we show their behaviour on a toy example. Specifically, data are generated using model (1.1), with $n = 100$, $p = 500$ and $\sigma = 4$. The p variables are drawn from multivariate standard normal distribution, and the first five variables have pairwise correlation $\rho = 0.9$ with each other, and the remaining variables are independent from each other, and independent from the first five correlated variables. There are 10 relevant variables, i.e., $s_0 = 10$. The five correlated variables have coefficients 1, and the first five independent variables have coefficients 6, while other variables have coefficients 0. Coefficients can be summarised as follows:

$$\beta_j = \begin{cases} 1 & \text{for} \quad j = 1, \ldots, 5 \\ 6 & \text{for} \quad j = 6, \ldots, 10 \\ 0 & \text{for} \quad j = 11, \ldots, 500 \end{cases} \tag{1.13}$$

Penalty level is chosen by 10-fold cross-validation, and for each method we present the number of selected variables and coefficient estimates for the relevant variables. The results are shown in Table 1.1.

	model size	$\hat{\beta}_1, \ldots \hat{\beta}_5$ (correlated relevant variables)	$\hat{\beta}_6, \ldots \hat{\beta}_{10}$ (independent relevant variables)
Ridge	500	0.29, 0.37, 0.45, 0.41, 0.41	0.73, 0.62, 0.66, 0.68, 0.52
Lasso	48	0.00, 0.00, 2.53, 1.72, 0.00	5.24, 5.41, 4.97, 5.33, 4.43
ENet	144	0.00, 0.59, 1.48, 1.21, 0.92	4.13, 3.71, 3.77, 3.79, 3.01
SCAD	8	0.00, 0.00, 0.00, 0.00, 4.30	5.88, 6.44, 5.43, 6.17, 4.87
Adalasso	20	1.29, 0.00, 0.00, 0.00, 2.62	5.25, 6.89, 4.69, 5.39, 5.71
Dantzig	41	0.00, 0.00, 2.78, 1.36, 0.00	5.26, 5.26, 4.88, 5.41, 4.80

Table 1.1: Model size and coefficient estimates for different methods in the toy example. See text for details of data generation. Model size and coefficient estimates of the five correlated relevant variables ($\beta_j = 1, j = 1, \ldots, 5$) and the five independent relevant variables ($\beta_j = 6, j = 6, \ldots, 10$) are presented. ENet stands for Elastic Net, and AdaLasso stands for Adaptive Lasso. α for Elastic Net is chosen to be 0.1 in this table.

1.4.2.1 Ridge Regression

When strong multicollinearity exists, i.e., some variables can be largely approximated by others linearly, $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is (nearly) singular, and the MLE is highly variable or not uniquely defined. Ridge Regression [Hoerl and Kennard, 1970] addresses this problem by adding an l_2 penalty, i.e., $P_\lambda(\boldsymbol{\beta}) = \lambda||\boldsymbol{\beta}||_2^2$.

Properties The l_2 penalty reduces variation of $\hat{\boldsymbol{\beta}}$ by introducing bias through the penalisation of coefficient magnitudes. As λ increases, the bias of the coefficient estimates increases while their variance decreases. It is equivalent to adding λ to the diagonal elements of $\mathbf{X}^T\mathbf{X}$, making it less close to being singular and therefore improving the stability of the matrix inversion.

With carefully chosen λ, the Ridge estimator outperforms the MLE with respect to prediction, especially when variables are highly non-orthogonal [Hoerl and Kennard, 1970]. The l_2 penalty results in Ridge Regression having a grouping effect, such that highly correlated variables tend to have similar coefficient estimates [Zou and Hastie, 2005]. From Table 1.1 the coefficient estimates for the five correlated relevant variables are similar to each other thanks to the grouping effect. Ridge Regression is also known to perform well in prediction when many variables are relevant with small coefficients, but is suboptimal when the underlying model is sparse [see e.g. Ogutu et al., 2012]. Indeed, from Table 1.1 its coefficient estimates of relevant variables are significantly underestimated compared to other methods.

Since the l_2 penalty is not singular at the origin, Ridge Regression does not directly perform variable selection. From Table 1.1 we see that Ridge Regression is the only method that does not perform variable selection, since all p coefficients have non-zero estimates. With an appropriate thresholding strategy [e.g., Shao and Deng, 2012], it can be seen as an alternative. Specifically, the authors proposed a threshold $a_n = Cn^{-\alpha}$ for some $C > 0$ and $0 < \alpha \leq 1/2$, such that variables with absolute coefficients larger than a_n are selected to be in the model. The thresholding enables Ridge Regression to achieve variable selection, and the authors proved that under certain sparsity assumptions, the thresholded Ridge Regression achieves consistency of variable selection, estimation and prediction.

Computation Ridge Regression has an analytical solution $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}+\lambda\mathbf{I})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}$.

1.4.2.2 Lasso

The Lasso (least absolute shrinkage and selection operator) estimator [Tibshirani, 1996] takes the form given in (1.9) with an l_1-norm penalty: $P_\lambda(\beta_j) = \lambda|\beta_j|$. This shrinks coefficients towards zero, with some set to exactly zero, and λ controls the amount of shrinkage and degree of sparsity.

Properties The theoretical properties of the Lasso have been well-studied and are mostly based on asymptotic analysis. We highlight some of the main properties here.

The design matrix $\mathbf{X}$ needs to satisfy the irrepresentable condition for consistent variable selection [Zhao and Yu, 2006]. This condition specifies a restriction on covariance between variables, such that the total amount of any irrelevant variables that can be represented by the relevant variables is limited, or in other words, irrelevant variables are irrepresentable by the relevant ones. More formally, for a scalar a, let sign(a) denote the sign of a, such that sign(a) = 1 if $a > 0$, -1 if $a < 0$, or 0 if $a = 0$. Additionally, for a vector $\mathbf{v}$, let sign($\mathbf{v}$) denote the vector of signs of the elements in $\mathbf{v}$. Then, the strong irrepresentable condition is satisfied if $||\Sigma_{\boldsymbol{\omega}^c\boldsymbol{\omega}}\Sigma_{\boldsymbol{\omega}\boldsymbol{\omega}}^{-1}\text{sign}(\boldsymbol{\beta}_{\boldsymbol{\omega}})||_\infty \leq 1-\tau$ for some $\tau > 0$, and the weak irrepresentable condition is satisfied if $||\Sigma_{\boldsymbol{\omega}^c\boldsymbol{\omega}}\Sigma_{\boldsymbol{\omega}\boldsymbol{\omega}}^{-1}\text{sign}(\boldsymbol{\beta}_{\boldsymbol{\omega}})||_\infty < 1$. Here, $\Sigma_{\boldsymbol{\omega_1},\boldsymbol{\omega_2}} = \frac{1}{n}\boldsymbol{X}_{\boldsymbol{\omega_1}}{}'\boldsymbol{X}_{\boldsymbol{\omega_2}}$, $\boldsymbol{\omega} = \{j : \beta_j \neq 0, 1 \leq j \leq p\}$, $\boldsymbol{\omega^c} = \{j : \beta_j = 0, 1 \leq j \leq p\}$, and $||\cdot||_\infty$ is the infinity norm. The irrepresentable condition does not hold when $\mathbf{X}$ is ill-posed with strong multicollinearity, which is common in high-dimensional data.

Let λ_n indicate that λ is a function of sample size n. Under certain regularity conditions with respect to the design, the following holds: if $\lambda_n/n \to 0$ and $\lambda_n/n^{\frac{1+c}{2}} \to \infty$ for some $0 \leq c < 1$, then the strong irrepresentable condition implies sign consistency [Zhao and Yu, 2006],

$$\lim_{n\to\infty} P(\text{sign}(\hat{\boldsymbol{\beta}}) = \text{sign}(\boldsymbol{\beta})) = 1, \tag{1.14}$$

which is stronger than selection consistency.

Bach [2008] proves that, under some regularity conditions with respect to the

design, let $\lambda_n = \frac{\lambda_0}{\sqrt{n}}$ for some $\lambda_0 > 0$, the probability of Lasso detecting all relevant variables grows to one in exponential rate with respect to n, while the probability of estimating an incorrect sign pattern converges to a limit in (0,1). Wainwright [2009] proves that consistent variable selection of Lasso requires

$$s_0 log(p) = o(n), \tag{1.15}$$

which establishes the relationship between n, p and s_0 for variable selection purpose.

Lasso can only select up to n variables, which is not ideal for small n, large s_0 problems. Finally, Lasso does not have group selection property, such that it tends to only select a few variables from a group of highly correlated variables, which is not ideal when many signals are correlated with each other. From Table 1.1 we see that Lasso only manages to pick up two out of the five strongly correlated relevant variables, which illustrates the fact that Lasso has difficulty discovering all the highly correlated relevant variables. When strong multicollinearity breaks the irrepresentable condition as in this case, Lasso fails to achieve consistent variable selection.

For prediction, Lasso can achieve consistency with respect to prediction loss under almost no assumptions [Chatterjee, 2013]. λ should typically be larger for consistent variable selection than for consistent prediction, and the prediction-optimal λ can lead to inclusion of many false positives [Meinshausen and Bühlmann, 2006]. From Table 1.1 we see that Lasso performs variable selection such that only 48 variables have non-zero estimates, although due to cross-validation getting λ for prediction, the model size is overly large.

Computation Lasso optimization can be solved with the generalised gradient descent approach [Friedman et al., 2010], which is implemented in the *glmnet* package. The Lasso problem can also be solved by the LARS algorithm with certain modification [Efron et al., 2004], for all $\lambda \in [0, \infty]$ (see Section 1.3.4). Specifically, in the LARS procedure, while updating $\hat{\boldsymbol{\mu}}$, whenever a non-zero coefficient hits zero, the corresponding variable is removed from the selected set of variables and the equiangular joint direction is updated. LARS can provide computational ef-

ficiency when the solution is investigated for many values of tuning parameter λ [Tibshirani and Taylor, 2011]. The level of regularization λ is usually determined to minimise cross-validation error in practice.

1.4.2.3 Elastic Net

Elastic Net [Zou and Hastie, 2005] is (1.9) with a convex combination of l_1 and l_2 penalty $P_\lambda(\beta_j) = \lambda_1|\beta_j| + \lambda_2\beta_j^2 = \lambda\left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2\right)$, where $\lambda_1, \lambda_2 > 0$ and $\alpha \in [0, 1]$ controls the balance of the two (α=1 and α=0 correspond to Lasso and Ridge Regression respectively).

Properties The l_1 penalty guarantees automatic variable selection and continuous shrinkage, while the l_2 penalty helps select correlated variables (signal or non-signal) simultaneously thanks to the group selection property, and it also stabilises the solution path. From Table 1.1 we see that Elastic Net selects more correlated variables than Lasso, and coefficient estimates for those correlated variables tend to be reasonably similar in magnitudes due to its group selection property. Its selection and estimation behaviour is in between that of Lasso and Ridge Regression, due to the compromise of l_1 and l_2 penalty. In the $p > n$ scenario, Elastic Net can select more than n variables, in contrast to a maximum of n variables for Lasso. In the toy example when $\alpha = 0.1$, Elastic Net selects 144 variables, a value larger than $n = 100$. Comparison of Elastic Net penalty and l_1, l_2 penalties is shown in Figure 1.2. The shape of the Elastic Net penalty is in between that of the l_1 and l_2 penalties. The edges are strictly convex for both Lasso and Elastic Net penalties, and singularities at the axes (where the contour hits the axes) guarantee their sparse solution.

Computation Elastic Net problem can be converted to Lasso problem with augmented data. Specifically, let $\boldsymbol{X}^* = (1+\lambda_2)^{-\frac{1}{2}}\begin{pmatrix}\mathbf{X}\\ \sqrt{\lambda_2}\mathbf{I}\end{pmatrix}$, and $\boldsymbol{Y}^* = \begin{pmatrix}\mathbf{Y}\\ \mathbf{0}\end{pmatrix}$, then $M(\lambda', \boldsymbol{\beta}^*) = ||\boldsymbol{Y}^* - \boldsymbol{X}^*\boldsymbol{\beta}^*||_2^2 + \lambda'||\boldsymbol{\beta}^*||_1$, and $\hat{\boldsymbol{\beta}}(\text{NEnet}) = \frac{1}{\sqrt{1+\lambda_2}}\hat{\boldsymbol{\beta}}^*$ is the naive elastic net solution. To correct extra bias introduced by double shrinkage of naive elastic net (Ridge-type shrinkage followed by Lasso-type shrinkage), scaling is necessary and elastic net solution is defined as $\hat{\boldsymbol{\beta}}(\text{Enet}) = (1+\lambda_2)\hat{\boldsymbol{\beta}}(\text{NEnet})$. With

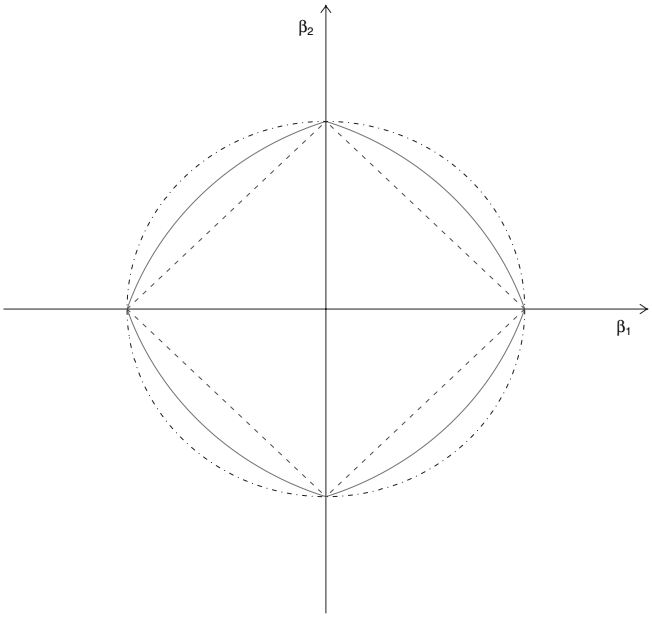


Figure 1.2: Two-dimensional contour plot to compare penalties for Lasso (. . . . .), Ridge Regression ($-\cdot-$) and Elastic Net (____, $\alpha = 0.5$) [Figure reproduced from Zou and Hastie, 2005]

modification of LARS, the above problem can be efficiently solved by the LARS-EN algorithm [Zou and Hastie, 2005].

1.4.2.4 SCAD

Lasso-type approaches introduce bias for large coefficients, since they equally penalise each variable. Smoothly Clipped Absolute Deviation (or SCAD) [Fan and Li, 2001] guarantees the solution to be sparse, continuous, and nearly unbiased for large coefficients. It uses the following penalty in (1.9):

$$P_\lambda(\beta_j) = \begin{cases} \lambda|\beta_j|\,, & if \quad |\beta_j| \leq \lambda \\ -\frac{|\beta_j|^2 - 2a\lambda|\beta_j| + \lambda^2}{2(a-1)}\,, & if \quad |\beta_j| \in (\lambda, a\lambda] \\ \frac{(a+1)\lambda^2}{2}\,, & if \quad |\beta_j| > a\lambda \end{cases} \tag{1.16}$$

where $a > 2$ and $\lambda > 0$.

Properties SCAD penalty is a non-convex, quadratic spline function by which small coefficients are shrunk towards zero with a Lasso penalty, while large coefficients are not penalised. The resulting estimator is, unlike Lasso, nearly unbiased for large coefficients. Fan and Li [2001] and Fan et al. [2004] also show that SCAD enjoys an oracle property (assuming some regularity conditions) – it is simultaneously consistent for variable selection and estimation, where the latter is as efficient (asymptotically) as the ideal case when the true model is known in advance. For further details on the properties of SCAD, see Fan and Lv [2010] and references therein.

From Table 1.1 we see that thanks to its adaptive penalty, SCAD penalises large coefficients ($\beta_j = 6$ for $j = 6, \ldots, 10$) less than Ridge Regression, Lasso and Elastic Net such that corresponding estimation is more accurate. Also its model size is significantly smaller than Lasso and Elastic Net, which is related to its oracle property for variable selection.

Computation The SCAD problem can be solved by an iterative procedure or one-step procedure [Fan and Li, 2001], with appropriately chosen initial estimators. We fix a to be 3.7, and tune λ by cross validation, which are recommended by Fan and Li [2001]. $a = 3.7$ minimises Bayes risks from an empirical point of view, but it can also be tuned via cross-validation.

1.4.2.5 Adaptive Lasso

Adaptive Lasso [Zou, 2006] also aims to adaptively impose penalties on different coefficients, such that large coefficients are less penalised than small ones. It uses the following penalty in (1.9)

$$P_\lambda(\beta_j) = \lambda w_j |\beta_j|, \tag{1.17}$$

where $w_j > 0$ is a known weight for variable j.

Properties When w_j has the form $\frac{1}{|\hat{\beta}_j|^\tau}$ for some $\tau > 0$ and $\hat{\beta}_j$ is a root-n consistent estimate of β_j (i.e., $\hat{\beta}_j - \beta_j = O_p(n^{-1/2})$), adaptive Lasso also enjoys the oracle property [Zou, 2006]. Coefficients of irrelevant variables are heavily

penalised since a large penalty will be placed for coefficients with small initial estimates; similarly, large coefficients tend to be un-penalised. From Table 1.1, we see that, compared to Lasso, Adaptive Lasso gives a more parsimonious model with 20 selected variables. It also achieves better estimation of large coefficients ($\beta_j = 6$) compared to Lasso and Elastic Net, which equally penalise all variables.

Computation Zou [2006] suggests to use least squares estimates as initial estimates, unless strong multicollinearity exists, in which case Ridge estimate is preferred due to its stability. Lasso estimate is also a common choice for the initial weights (for example, see Krämer et al. [2009]), since it is root-n consistent [Knight and Fu, 2000]. We use Lasso estimates as the initial estimates for adaptive Lasso throughout the thesis. The adaptive Lasso problem can be converted to a standard Lasso problem, which can then be solved by the LARS algorithm. In particular, let $\boldsymbol{x_j^*} = \frac{\boldsymbol{x_j}}{w_j}$, then the Lasso problem $\hat{\boldsymbol{\beta}}^* = \arg\max_{\boldsymbol{\beta}^*} ||\mathbf{Y} - \sum_{j=1}^{p} \boldsymbol{x_j^*} \beta_j^*||_2^2 + \lambda ||\boldsymbol{\beta}^*||_1$ can be solved to give the adaptive Lasso estimate $\hat{\beta}_j(\text{AdaLasso}) = \frac{\hat{\beta}_j^*}{w_j}, j = 1 \dots p$.

1.4.2.6 Dantzig Selector

The Dantzig Selector estimator [Candes and Tao, 2007] takes a different form to that in (1.1), namely:

$$\hat{\boldsymbol{\beta}}_\lambda = \underset{\boldsymbol{\beta}}{\text{argmin}} \left\{ \|\boldsymbol{\beta}\|_1 : \|\mathbf{X}^T (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) \|_\infty \leq \lambda \right\}, \tag{1.18}$$

Properties The Dantzig Selector and the Lasso are closely connected as discussed in Bickel et al. [2009], and under certain conditions on $\mathbf{X}$, Lasso and Dantzig Selector provide the same solution [James et al., 2009; Meinshausen et al., 2007]. From Table 1.1 we see in the toy example, Dantzig Selector and Lasso have similar model size and similar coefficient estimates for relevant variables.

It is proved that, in noiseless case where $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}$, if the design matrix $\mathbf{X}$ obeys the uniform uncertainty principle (UUP), Dantzig Selector can achieve exact recovery with some regularity assumptions on design. Loosely speaking, the UUP requires that any subset of columns of $\mathbf{X}$ with cardinality smaller than s_0 can be roughly seen as an orthogonal system. The oracle estimator achieves estimation

error $E||\boldsymbol{\beta}(\text{oracle}) - \boldsymbol{\beta}||_2^2 \geq \frac{1}{2} \times \sum_i \min(\beta_i^2, \sigma^2)$, and Dantzig Selector can achieve this ideal estimation error within a factor of $\mathcal{O}(\log p)$.

Computation The Dantzig Selector problem can be solved with linear program.

1.4.2.7 Other methods

There are numerous other penalised regression methods for high-dimensional data. For example, the minimax concave penalty (MCP) [Zhang, 2010] also achieves sparsity, continuity and near unbiasedness for large coefficients like SCAD. It minimises maximum concavity of penalised loss function, with restrictions on unbiasedness and variable selection. There are also penalised regression methods designed for specific types of data. When variables can be sorted into K groups, with p_k the size of group k ($k = 1 \dots K$), group Lasso [Yuan and Lin, 2006] uses $P_\lambda(\boldsymbol{\beta}) = \lambda \sum_{k=1}^{K} \sqrt{p_k} ||\boldsymbol{\beta^{(k)}}||_2$, where $\boldsymbol{\beta^{(k)}}$ are the coefficients of variables in group k. Coefficients from each group share the same level of penalty. To introduce sparsity within a group, Friedman et al. [2010] suggest adding an l_1 term into the group Lasso penalty. Fused Lasso [Tibshirani et al., 2005] is desirable for problems where variables can be ordered meaningfully. The penalty is $P_\lambda(\boldsymbol{\beta}) = \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=2}^{p} |\beta_j - \beta_{j-1}|$ for some $\lambda_1, \lambda_2 > 0$. The l_1 penalty on successive differences of coefficients introduces corresponding sparsity, and consequently local constancy of estimation. This is desirable for interpretation in certain scientific problems. There are also variants of methods aiming to improve the selection and predictive performance. For example, Zeng and Xie [2012] add an l_2 penalty to SCAD to obtain the group selection property, while preserving unbiasedness, continuity, and sparsity. The authors showed that this approach stabilises variable selection in multicollinearity settings, with enhanced prediction accuracy.

1.5 Bayesian linear regression

1.5.1 Bayesian inference

Bayesian inference is based on Bayes' theorem, and uses probability distributions to explicitly quantify the uncertainty in inference. Bayesian methods assume parameters to be unknown random variables instead of fixed values, and assign prior distributions to parameters, enabling prior knowledge to be specified. A full probability model is set up with a joint distribution for observed and unobserved quantities, and a posterior distribution of unobserved quantities of interest is obtained, conditional on the observed data. Inference is carried out using these posterior distributions.

Formally, let Θ be the unknown parameters of interests, and D the observed data. A prior distribution for Θ, $p(\Theta)$ quantifies prior information on the parameters before observing the data, and $p(D|\Theta)$ is the likelihood function. The posterior distribution of Θ given D is then calculated as

$$p(\Theta|D) = \frac{p(D|\Theta)p(\Theta)}{p(D)}, \tag{1.19}$$

where $p(D)$ is the marginal likelihood of the observed data, marginalised over Θ,

$$p(D) = \int_{\Theta} p(D|\Theta)p(\Theta)d\Theta. \tag{1.20}$$

The distribution of Θ can have its own parameter(s) τ, called hyperparameter(s), i.e., $\Theta \sim p(\Theta|\tau)$. τ can be a tuning parameter set by the user or can also be considered as a random variable and have its own prior.

1.5.2 Bayes factor

The Bayes factor is widely used for Bayesian model comparison. It is the ratio of the marginal likelihood of two competing models. Unlike classical likelihood ratio where parameters are maximised under corresponding models, they are integrated out. The Bayes factor quantifies the relative evidence of the competing models, and facilitates model selection.

Formally, in a model selection problem based on the observed data D, where one needs to choose from two candidate models M_1 and M_2, with parameters Θ_1 and Θ_2 respectively, the Bayes factor is defined as

$$BF = \frac{p(D|M_1)}{p(D|M_2)} = \frac{\int_{\Theta_1} p(D|\Theta_1, M_1)p(\Theta_1|M_1)d\Theta_1}{\int_{\Theta_2} p(D|\Theta_2, M_2)p(\Theta_2|M_2)d\Theta_2}, \tag{1.21}$$

where $p(M_1)$ and $p(M_2)$ are the prior probabilities of M_1 and M_2. The Bayes factor is the ratio of marginal likelihood of two model specifications, taking into account the uncertainty in parameters, in contrast to classical likelihood ratio,

$$LR = \frac{l(\hat{\Theta}_1|D, M_1)}{l(\hat{\Theta}_2, |D, M_2)}, \tag{1.22}$$

which is the ratio of two likelihood functions, maximised with respect to their parameters, and $\hat{\Theta}_1$ and $\hat{\Theta}_2$ are MLEs.

When M_1 and M_2 are equally likely a priori, the Bayes factor can also be simplified as $BF = \frac{p(M_1|D)}{p(M_2|D)}$, which is the ratio of posteriors of the two models. $BF > 1$ means model M_1 is more supported by the data than M_2, while $BF < 1$ means model M_2 is more supported. The Bayes factor does not depend on specific values of the parameters, and a sparsity prior for Θ can be incorporated into the Bayes factor to penalise complex models.

BIC can be used to approximate the Bayes factor [Kass and Raftery, 1995]. Specifically in linear regression, let M_2 and M_2 be two different models, then with certain priors on β's, the Bayes factor of model M_1 versus M_2 can be approximated as

$$BF_{12} \approx \frac{P_{BIC}(D|M_1)}{P_{BIC}(D|M_2)} = \exp(\frac{\Delta BIC_{21}}{2}), \tag{1.23}$$

where $BIC(M_i) = -2\text{log}l(\hat{\boldsymbol{\beta}}^i|D, M_i) + k_i\text{log}n$, $\hat{\boldsymbol{\beta}}^i$ and k_i being the MLE and degree of freedom of model M_i; $P_{BIC}(M_i) = \exp(-\frac{BIC(M_i)}{2})$; and $\Delta BIC_{21} = BIC(M_2) - BIC(M_1)$ [Wagenmakers, 2007].

1.5.3 Bayesian linear model

In a Bayesian linear regression model, $\boldsymbol{\beta}$ and σ^2 in (1.1) are treated as unknown random variables, and inferences are made based on the posterior distribution of $\boldsymbol{\beta}$ and σ^2 given the data. Let $\Theta = (\boldsymbol{\beta}, \sigma^2)$ and $p(\Theta)$ denote the prior distribution over Θ. Then the posterior distribution over Θ is

$$p(\Theta|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X}, \Theta)p(\Theta)}{p(\mathbf{Y}|\mathbf{X})}, \tag{1.24}$$

where

$$p(\mathbf{Y}|\mathbf{X}, \Theta) = N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}). \tag{1.25}$$

Since the posterior is a density instead of a point estimate, the level of associated uncertainty can be quantified using, for example, credible intervals.

In order to calculate the posterior probability, one must calculate $p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{X}, \Theta)p(\Theta)d\Theta$, which is not available in closed form in general. If that is the case, numerical approaches are required to evaluate the integral, which can be computationally intensive. The exception is when the prior is conjugate, which means that the prior and posterior of Θ have the same parametric form. Conjugate priors result in a closed-form solution for the integral and provide computation efficiency. Specifying the prior distribution for Θ is a challenging task, and a poor choice of prior distribution can lead to inaccurate posterior distribution, and consequently inferior model performance, especially when the sample size is small.

In Bayesian linear regression, a common conjugate prior for Θ is called the normal inverse gamma (NIG) distribution. It assumes $\boldsymbol{\beta}|\sigma^2$ follows a multivariate normal distribution, and σ^2 follows an inverse-gamma distribution. Formally, the joint distribution of $\boldsymbol{\beta}$ and σ^2 is

$$p(\boldsymbol{\beta}, \sigma^2) = p(\boldsymbol{\beta}|\sigma^2)p(\sigma^2) = N(\boldsymbol{\beta}|\boldsymbol{\mu}, \sigma^2\mathbf{V})IG(\sigma^2|a, b), \tag{1.26}$$

where $\boldsymbol{\mu}, \mathbf{V}, a > 0, b > 0$ are hyperparameters, and the inverse-gamma distribution $IG(\sigma^2|a, b)$ has the form

$$IG(\sigma^2|a, b) = \frac{b^a}{\Gamma(a)}(\sigma^2)^{-(a+1)}exp(-\frac{b}{\sigma^2}), \tag{1.27}$$

where $\Gamma(.)$ is the Gamma function. Due to conjugacy, applying Bayes' theorem (1.24), the joint posterior of $(\boldsymbol{\beta}, \sigma^2)$ is also a NIG distribution,

$$p(\boldsymbol{\beta}, \sigma^2|\mathbf{Y}, \mathbf{X}) = NIG(\boldsymbol{\mu}^\star, \mathbf{V}^\star, a^\star, b^\star), \tag{1.28}$$

where

$$\begin{aligned} \boldsymbol{\mu}^\star &= (\mathbf{V}^{-1} + \mathbf{X}^T\mathbf{X})^{-1}(\mathbf{V}^{-1}\boldsymbol{\mu} + \mathbf{X}^T\mathbf{Y}) \\ \mathbf{V}^\star &= (\mathbf{V}^{-1} + \mathbf{X}^T\mathbf{X})^{-1} \\ a^\star &= a + n/2 \\ b^\star &= b + \frac{1}{2}(\boldsymbol{\mu}^T\mathbf{V}^{-1}\boldsymbol{\mu} + \mathbf{Y}^T\mathbf{Y} - (\boldsymbol{\mu}^\star)^T(\mathbf{V}^\star)^{-1}\boldsymbol{\mu}^\star). \end{aligned} \tag{1.29}$$

σ^2 or $\boldsymbol{\beta}$ can be integrated out from the joint distribution to obtain marginal distribution of the other. A point estimate of parameters can be obtained by summarising the distribution, and the distribution explicitly reveals the uncertainty of the parameters.

1.5.4 Bayesian Ridge Regression

Ridge Regression can deal with $p > n$ scenarios, and is computationally efficient due to its closed-form solution. Ridge Regression can be formulated in a Bayesian way. Specifically, the Bayesian representation of Ridge Regression is as follows:

$$\begin{aligned} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\beta}, \sigma^2) &= N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}) \\ \boldsymbol{\beta}|\sigma^2, \lambda &\sim N\Big(\mathbf{0}, \frac{\sigma^2}{\lambda}\mathbf{I}\Big) \\ \sigma^2 &\sim IG(a, b), \end{aligned} \tag{1.30}$$

where $\lambda > 0$ is a tuning parameter controlling the degree of shrinkage. Note that this prior specification of σ^2 and $\boldsymbol{\beta}$ is the NIG prior (1.26) for linear model with $\boldsymbol{\mu} = \mathbf{0}$ and $\mathbf{V} = \frac{1}{\lambda}\mathbf{I}$. Applying Bayes's theorem (1.24) and integrating out σ^2, the posterior marginal distribution of $\boldsymbol{\beta}$ follows a multivariate Student t-distribution

[Denison et al., 2002]

$$p(\boldsymbol{\beta}|\mathbf{X},\mathbf{Y}) = MVSt_{2a+n}(\hat{\boldsymbol{\beta}}, \frac{b+\frac{1}{2}(\boldsymbol{Y}^{\mathrm{T}}\mathbf{Y} - \hat{\boldsymbol{\beta}}^{\mathrm{T}}(\lambda\mathbf{I}+\mathbf{X}^{\mathrm{T}}\mathbf{X})\hat{\boldsymbol{\beta}})}{a+\frac{n}{2}}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}), \tag{1.31}$$

where $MVSt_\nu(\boldsymbol{\mu},\boldsymbol{\Sigma})$ has probability density function

$$p(\mathbf{x}) = \frac{\Gamma[(\nu+p)/2]}{\Gamma(\nu/2)\nu^{p/2}\pi^{p/2}|\boldsymbol{\Sigma}|^{1/2}}[1+\frac{1}{\nu}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})]^{(\nu+p)/2}, \tag{1.32}$$

and p is the dimension of $\mathbf{x}$. The mean $\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$ is identical to frequentist Ridge Regression estimate in Section 1.4.2.1. The hyperparameters a and b for the prior on σ^2 are typically chosen to be non-informative, i.e., a, b are close to zero, and as in frequentist Ridge Regression, the posterior distribution of β's is shrunk towards zero. The marginal distribution of $\mathbf{Y}$ also follows a multivariate Student t-distribution [Denison et al., 2002], i.e.,

$$p(\mathbf{Y}) = \int P(\mathbf{Y}|\boldsymbol{\beta},\sigma^2)P(\boldsymbol{\beta},\sigma^2)d\boldsymbol{\beta}d\sigma^2 = MVSt_{2a}(\mathbf{0}, \frac{b}{a}(\mathbf{I}+\frac{\mathbf{X}\mathbf{X}^T}{\lambda})) \tag{1.33}$$

1.6 Ensemble learning methods

Ensemble learning aims to improve machine learning methods by combining different models in a strategic way. While ensemble learning requires more computations than a single model, it can obtain better model performance than any of the constituent models alone. It is known that diversity among candidate models in ensemble learning helps yield better results, and many ensemble learning methods seek to increase such diversity [Brown et al., 2005]. Some common ensemble learning methods include bootstrap aggregation [bagging, Breiman, 1996], boosting [Freund and Schapire, 1999] and stacking [Wolpert, 1992]. Below we give details on some ensemble learning methods, and their applications.

1.6.1 Bootstrap aggregation

Bootstrap aggregation (bagging) is one of the most intuitive ensemble based methods, where diversity of models (i.e., differences among models) is obtained by bootstrapping the training data. That is, new training data samples of size n are randomly drawn with replacement from the full data. For large n, each bootstrap sample is expected to have $\sim 63.2\%$ of the unique samples of the full data, while the rest are duplicates. Models are then fitted to each bootstrap sample before they are combined through averaging the output. Each model has a high variance and low bias, and combining them reduces the variance. According to Breiman [1994, 1996], bagging can improve the instability in procedures such as classification and regression trees, and variable selection in linear regression.

Bootstrap-enhanced Lasso

An interesting attempt to combine bagging and penalised linear regression is called Bootstrap-enhanced Lasso, or Bolasso for short [Bach, 2008], which runs Lasso on several bootstrap samples of the training data, and the intersect of the selected variables of bootstraped Lasso are defined as the relevant variables. Under certain conditions, Bolasso leads to consistent model selection, and in general, Bolasso can fix the instability of Lasso and correctly retrieve the sparsity pattern when Lasso's consistent selection conditions are not satisfied. Bolasso enhances Lasso's variable selection performance and is consistent under more general assumptions.

Stability Selection

Bolasso is mainly designed for low-dimensional data, and its properties are proved under the condition that p is fixed with respect to n. Stability Selection [Meinshausen and Bühlmann, 2010] is a very general method combining subsampling and variable selection methods, with the aim of providing finite sample false positive error control. Specifically, M random data subsamples of size $\tilde{n} < n$ are generated by sampling without replacement. $\tilde{n}$ is typically chosen to be $\lfloor n/2 \rfloor$ or $\lfloor 0.632n \rfloor$, which closely resemble the bootstrap. Applying a variable selection procedure, with regularization parameter λ, to these datasets gives a score $\hat{\Pi}_{\lambda,j}$ indicating the frequency with which variable j is selected among the M iterations, i.e., a selection

probability. Let Λ denote the set of considered values for the regularization parameter. Then, a set of "stable variables" is obtained by choosing those variables that have selection probabilities larger than a cutoff value $\pi_{\text{thr}} \in (0, 1)$ for any $\lambda \in \Lambda$.

In contrast to the penalised regression methods and Bolasso, Stability Selection does not require setting of the parameter λ, but instead requires the cutoff π_{thr} to be chosen (Bolasso fixes π_{thr} to be 1). Meinshausen and Bühlmann [2010] provide theoretical results showing how π_{thr} can be chosen to achieve a user-specified upper bound $\tilde{V}$ on the expected number of false positives $\mathbb{E}[V]$, assuming a fixed set of regularization parameters Λ. Alternatively, the user can fix π_{thr} and then the theory shows how Λ should be chosen to achieve the desired upper bound on $\mathbb{E}[V]$. In this thesis we use the Lasso as the variable selection procedure with Stability Selection.

1.6.2 Random forest

Decision tree

A decision tree seeks to predict the unseen response based on input variables. It can be presented by a tree structure consisting of nodes and branches. The decision tree model is built in a structured way, where the data samples are segmented into subgroups through a series of decisions, and each internal node corresponds to a test on an input variable. The model starts with all samples in one node, called the root node, and the first split is made with one variable, such that samples are partitioned among its child nodes, according to the decision rule for the variable. Branches connecting an internal node and its child nodes represent the decision rule outcomes. Then child nodes follow similar procedure to further segment the data recursively, until leaf nodes are reached where further splits are not possible or will not improve the model fit. At each split, the model fit is evaluated by homogeneity in class labels/observed responses of resulting child nodes, and one chooses the split that increases such homogeneity. A leaf node represents a class label/observed response or a distribution of the class labels/observed responses.

Greedy approach is typically applied for construction of a decision tree. The most significant variable is chosen to be the root node, and more significant vari-

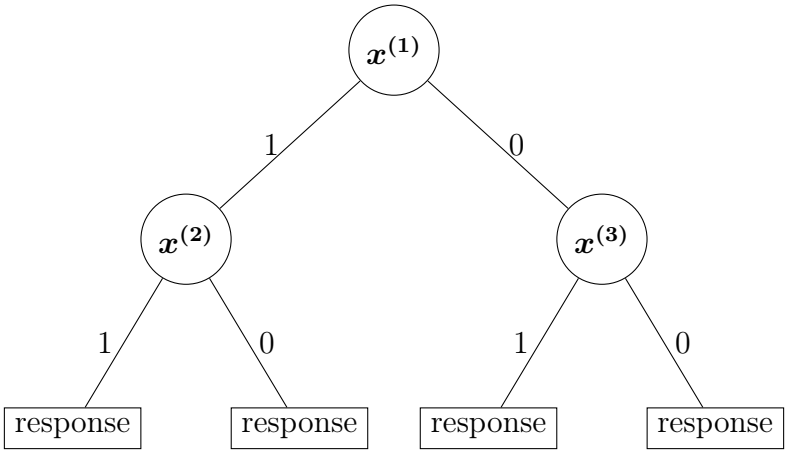


Figure 1.3: An illustration of decision tree

ables are closer to the root node. The significance of a variable can be calculated using information theory criteria (for example, a measure of node impurity decrease). Prediction of class label for new data is made based on the paths of decisions from the root node to leaf nodes.

Figure 1.3 illustrates the structure of a decision tree. $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}$ and $\boldsymbol{x}^{(3)}$ are binary variables taking values zero and one, and they are ranked by their significance. The node at top is the root node, and represents a test on $\boldsymbol{x}^{(1)}$ being zero or one. A split is made to represent two different outcomes of this test. Depending on the test result, one tests the next variable ($\boldsymbol{x}^{(2)}$ or $\boldsymbol{x}^{(3)}$) and further split the data, reaching the leaf nodes at bottom representing the observed responses. At each split, the goal is to increase the homogeneity in responses among child nodes.

Classification trees and regression trees are two common types of decision tree, where predicted outcome is categorical for classification trees and continuous for regression trees.

Random forest

Random forests [Breiman, 2001] are an ensemble learning method designed for classification and regression, among other tasks. The idea is to construct a series of decision trees, and the output is an average of the individual decision trees. The random forest method is known to correct the overfitting problem of the ordinary

decision tree method [Hastie et al., 2008]. Each individual tree may be subject to increased bias in estimation and less interpretability, but the final model in general improves method performance significantly.

Random forest involves subsampling of samples and variables. B trees are constructed with B bootstrapped training data, and to further decorrelate the trees, at each split of an individual decision tree, a sample of $m < p$ randomly chosen input variables are considered as split candidates. By doing so, variables have a more uniform chance to be selected, instead of most trees selecting the same set of strong variables, causing trees to be highly correlated, with high variance. Note that if we choose $m = p$, random forest is the same as bagging, and using a small value of m can be useful when strong multicollinearity exists.

1.6.3 Random Lasso

Random Lasso [Wang et al., 2011] is a variable selection method motivated by random forest, aiming to solve some of the issues of Lasso. Specifically, Lasso does not have group selection property under strong multicollinearity, especially when coefficients of correlated variables have different magnitudes or signs [Wang et al., 2011]; Lasso has a limit on the number of selected variables, which could be undesirable when a large number of variables are relevant.

Random Lasso consists of two steps and can be summarised as follows:

Step 1. Calculate importance measure

1a. For $b_1 = 1, \dots B$, do:

(i) Draw a bootstrap sample by sampling with replacement from full data

(ii) Randomly select q_1 variables from the bootstrapped samples and apply Lasso to these variables to obtain coefficient estimates $\tilde{\beta}_j^{(b_1)}, j = 1 \dots p$. Coefficients for those not among the q_1 variables are set to be zero

1b. The importance measure of x_j is calculated as $I_j = |\frac{\sum_{b_1=1}^{B} \tilde{\beta}_j^{(b_1)}}{B}|, j = 1 \dots p$

Step 2. Select variables

2a. For $b_2 = 1, \ldots B$, do:

(i) Draw a bootstrap sample by sampling with replacement from full data

(ii) Randomly select q_2 variables from the bootstrapped samples, with selection probability of x_j proportional to its importance measure I_j. Apply Lasso to these variables to obtain coefficient estimates $\hat{\beta}_j^{(b_2)}, j = 1 \ldots p$. Coefficients for those not in q_2 variables are set to be zero

2b. The final estimator is $\hat{\beta}_j = \frac{\sum_{b_2=1}^{B} \hat{\beta}_j^{(b_2)}}{B}$. Wang et al. [2011] suggest to set the threshold on magnitudes of coefficient estimates to be $1/n$, such that those variables with $|\hat{\beta}_j| < 1/n$ are removed from the model.

Sampling of variables in Step 1a and 2a breaks down the correlation structure, such that highly correlated variables are selected in different candidate models. In penalised linear regression choosing the right amount of penalisation is notoriously difficult, especially for high-dimensional data, and there may not be a single penalty level that can perfectly recover the sparsity pattern [Meinshausen and Bühlmann, 2010]. Random Lasso applies Lasso on different candidate models, with different penalty levels; the results are summarised across candidate models, with more stable parameter estimation. The number of selected variables is no longer limited by n. q_1 and q_2 are tuning parameters, and Wang et al. [2011] suggest using cross-validation to find optimal q_1 and q_2, by repeated application of Random Lasso, which can be time-consuming.

1.7 Performance metrics

Prediction. To assess predictive performance we introduce two metrics. In Chapters 2 and 3, predictive performance is compared across different scenarios. In any simulation scenario, we generate a test data of size n_{test}, and compare the predicted response $\hat{\boldsymbol{y}}$ with the observed response of test data using root mean squared

error (RMSE), calculated as

$$\text{RMSE} = \|\mathbf{y}_{\text{test}} - \hat{\boldsymbol{y}}\|_2/\sqrt{n_{\text{test}}}, \tag{1.34}$$

where $\hat{\boldsymbol{y}} = \mathbf{X}_{\text{test}}\hat{\boldsymbol{\beta}}$, $\mathbf{y}_{\text{test}}$ and $\mathbf{X}_{\text{test}}$ are the test responses and design matrix respectively, and $\hat{\boldsymbol{\beta}}$ are the coefficient estimates obtained using the training data. This metric implicitly involves the impact of error terms, which is important when comparing among datasets with different noise levels. In Chapter 4 and 5, predictive performance is evaluated within each scenario, and we use mean squared error (MSE) as defined in Tibshirani [1996], i.e., MSE$=(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^{\text{T}}\mathbf{V}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})$, where $\mathbf{V}$ is the population covariance matrix of $\mathbf{X}$. This metric excludes the influence of error terms, such that the contrast of prediction performance is clearer among methods.

Variable selection. For assessment of variable selection, we use true positive rate (TPR) and positive predictive value (PPV)

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \in [0, 1] \qquad \text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}} \in [0, 1]. \tag{1.35}$$

where TP, FP, and FN are the number of true positives, false positives, and false negatives respectively, with respect to the true underlying model $\boldsymbol{\gamma}$. We further define false positive rate (FPR) as

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \in [0, 1], \tag{1.36}$$

where TN is the number of true negatives.

Variable ranking. For ranking, we assess performance using the partial area under the receiver operating characteristic curve (pAUC). A receiver operating characteristic (ROC) curve shows the performance of binary classification as the discrimination threshold varies. Specifically, after ranking p variables from the most important to the least important, take the top k important variables and calculate FPR and TPR. The ROC curve is TPR values plotted against FPR values as k varies from 1 to p. The area under the ROC curve (AUC) quantifies how well the variables are ranked, and larger AUC indicates better ranking. pAUC is

the (rescaled) AUC score with the restriction on FPR to be smaller than certain threshold. Both AUC and pAUC are between 0 and 1. We use pAUC to evaluate ranking in Chapter 2 and 3, since for high-dimensional data we are typically interested in those high-ranked variables.

1.8 Thesis overview

The thesis is organised as follows. In Chapter 2 we present a large-scale comparison of high-dimensional regression methods, where we systematically consider the influence of various factors on several widely-used methods. In Chapter 3, we extend the simulations in Chapter 2 to investigate some further questions with respect to model assumptions and parameter tuning. In Chapter 4 we propose a novel variable selection method called STructural RANDomised Selection (STRANDS), which follows the spirit of random Lasso. It uses repeated subsampling of variables to reduce dimensionality before variable selection, and takes into account the correlation structure of the data. In Chapter 5 we propose another method combining Bayesian Ridge Regression and forward selection, to improve the prediction performance of Ridge Regression in sparse settings. Finally, in Chapter 6 we summarise and discuss the main conclusions, and point out potential directions for further study.

Chapter 2

Systematic comparison of penalised linear regression methods

In this Chapter we present a large-scale simulation study systematically comparing six popular penalised regression methods in high-dimensional settings. We assess their comparative performance with respect to ranking, prediction and variable selection, using data with various characteristics. In Chapter 3 we extend this comparison study to explore additional topics with respect to model assumptions and parameter specifications.

2.1 Introduction

In a wide range of applications it is now routine to encounter regression problems where the number of variables p exceeds the sample size n, often greatly. Even in the simple case of linear models with independent Gaussian noise, estimation is nontrivial and requires specific assumptions. A common and often appropriate assumption is that of sparsity, where only a subset of the variables have non-zero coefficients, with the number s_0 of such relevant variables usually assumed much smaller than p.

Penalised methods augment the regression log-likelihood with a penalty term that encodes a structural assumption such as sparsity (see Section 1.4 for details). Recent years have seen much progress in theory and methodology for penalised

regression (see Bühlmann and van de Geer, 2011, for a lucid account). However, while the theoretical developments have been remarkable and insightful, they cannot go quite as far as telling the user which method to use in a given finite-sample setting. Even with finite-sample theories, deducing relative performance between multiple methods is challenging. Meanwhile, rapid methodological progress has meant a wide range of plausible approaches to choose between.

In this chapter we aim to fill this gap via a systematic empirical comparison of a number of penalised regression methods, which could guide users towards selecting methods for specific applications. We consider six popular methods (Lasso, Elastic Net, Ridge Regression, SCAD, Dantzig Selector and Stability Selection) and 1,863 data-generating scenarios. It is obvious that large departures from modeling assumptions can produce poor results. Here our intention is not so much to look at robustness to such departures, but rather to look at variation in performance even in the favourable case where assumptions broadly hold. In this way we limit the scope of our simulation study, and focus only on properties of the methods, rather than their model assumptions.

In the simulations, we vary a number of factors in a relatively fine-grained manner within an essentially full factorial design (i.e. all combinations of factors). In addition to the main simulations, we also compare methods using semi-synthetic data (real variables but simulated responses) to study the generalisability of our findings.

Our main findings are: (i) we find substantial variation in performance between methods with no unambiguous winner across scenarios (i.e details of the data-generating set-up matter) and this is despite the fact that we focus on a relatively narrow class of models broadly favourable to the methods employed, (ii) relative performance depends on the specific goals, (iii) Lasso is relatively stable in the sense that it performs competitively in many of the scenarios considered here, and (iv) an l_2 penalty is beneficial only in extreme correlation scenarios. In addition, we find evidence of an interesting "phase transition"-like behavior for SCAD, where it goes from being the best performing method to the worst as scenario difficulty increases. Our results and code also provide a resource, allowing users to compare the methods considered here against each other across many scenarios and also to extend the study with other (existing or novel) methods.

A number of previous papers have examined the empirical performance of penalised regression methods. Meinshausen and Bühlmann [2010] consider large p problems from a selection perspective. Bühlmann and Mandozzi [2014] is a more comprehensive study using 8 semi-synthetic datasets and 128 scenarios. They evaluate screening (i.e., how well a selected set of variables contains the relevant ones) and ranking properties in high-dimensional settings. In contrast to previous work, our design is more complete and systematic. We use finer grids on factors including n, p, s_0 and signal-to-noise ratio (SNR) so that our results cover a wider range of designs, allowing us to more fully investigate the trends in relative performance. We also consider several multicollinearity parameters, so we can better understand this practically important factor. Furthermore, we evaluate all three of prediction, selection and ranking, using specific performance metrics for each. To limit scope we do not consider Bayesian methods here but note that there have been some interesting empirical comparisons of frequentist and Bayesian methods [including Bondell and Reich, 2012; Celeux et al., 2012]. To the best of our knowledge, the present work is more comprehensive in terms of data-generating scenarios and metrics than previous work.

The remainder of the Chapter is organised as follows. In Section 2.2, we describe our simulation strategy, including the data-generating factors considered. We also give details of how the methods are implemented and the performance metrics used. Section 2.3 presents the results from our main simulation study. Results from semi-synthetic data, based on a cancer study, appear in Section 2.4. We conclude with a discussion in Section 2.5.

2.2 Methods

In this section we use model (1.1), $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, to generate high dimensional data ($n < p$), and compare relative performance of Lasso, Elastic Net, Ridge Regression, SCAD, Dantzig Selector and Stability Selection. Details of these methods can be found in Section 1.4.

2.2.1 Simulation set-ups

We set $\boldsymbol{\beta}$ to have s_0 non-zero entries (all set to 3) with σ then set to obtain a desired SNR, defined here as SNR $= \sqrt{\boldsymbol{\beta}^T \mathbf{X}^{\mathrm{T}} \mathbf{X} \boldsymbol{\beta}/(n\sigma^2)}$. We acknowledge that having homogeneous coefficients is unrealistic for real life applications; the purpose of our design is to limit the scope of simulations. We consider heterogeneous coefficients in Section 3.2.

We consider the following three designs:

- **Independence design.** All p variables are i.i.d. standard normal.
- **Pairwise correlation design.** The p variables are partitioned into B blocks, each of size $p^B = p/B$. All variables are standard normal but with correlation between any pair of variables within the same block set to ρ. Variables in different blocks are independent of each other. The number of relevant variables within a block is s_0^B for the first s_0/s_0^B blocks, with the remaining blocks containing no relevant variables.
- **Toeplitz correlation design.** As for pairwise correlation, but with variables $\mathbf{x}_{j_1}$ and $\mathbf{x}_{j_2}$ within the same block having correlation $0.95^{|j_1 - j_2|}$. We only consider two relevant variables per block, $s_0^B = 2$,with their positions, j_1' and j_2', within a block chosen such that $|j_1' - j_2'| = 7$, to give a correlation of $0.95^7 \approx 0.7$.

We consider the effects of n, p, s_0, SNR, ρ, p^B, s_0^B and correlation design in a systematic way via 1,863 simulation scenarios, each corresponding to a different configuration. The values considered for each factor are shown in Table 2.1 and we cover all combinations of the factors with the following exceptions: for Toeplitz correlation design, we consider only $p^B = 100$ with two signals per block ($s_0^B = 2$); and for correlation designs we exclude some combinations of s_0^B and $B = p/p^B$ which violate the necessary constraint $s_0^B \geq s_0/B$ (see Table 2.2).

The parameter range is chosen such that we are able to gain insights with reasonable computational times. Specifically, there should be enough scenarios where differences of method performance can be observed; when scenario is either too 'easy' or too 'difficult', methods tend to have similar performance. We are

aware that the parameter range can be unrealistic for many studies. For example, n can be much larger than values considered here in GWAS studies, and p can be tens of thousands or even millions for many omics assays. The goal is not to mimic the realistic scenarios, but rather to depict the trend of method performance with respect to different parameters, which allows extrapolation for parameters out of the range considered.

	Factors	Values considered
Independence and correlation designs	Sample size, n	100, 200, 300
	Dimensionality, p	500, 1000, 2000, 4000
	Sparsity, s_0	10, 20, 40
	Signal-to-noise ratio, SNR	1, 2, 4
Correlation designs only	Pairwise correlation within a block, ρ	0.5, 0.7, 0.9
	Toeplitz correlation within a block	$0.95^{\lvert j_1 - j_2 \rvert}$ for $\mathbf{x}_{j_1}$, $\mathbf{x}_{j_2}$ in same block
	Block size, p^B	10, 100
	Number of signals per block, s_0^B	1, 2, 5

Table 2.1: Factors varied in the simulation study and values considered. Note that for the correlation design, the s_0^B signals per block applies to the first s_0/s_0^B blocks only.

2.2.2 Method implementations

Tuning parameters are set to reflect the way methods would typically be used by practitioners. For Lasso, Elastic Net, Ridge Regression, SCAD and Dantzig Selector, λ is set via 10-fold cross-validation. Following Bühlmann and Mandozzi [2014], we implement two versions of Elastic Net with $\alpha = 0.3$ and $\alpha = 0.6$, referred to as heavy Elastic Net (HENet) and light Elastic Net (LENet) respectively (i.e., HENet has higher weight of l_2 penalty than LENet). For SCAD, we set $a = 3.7$, as recommended by Fan and Li [2001]. For Stability Selection, we set the number of iterations to M=100 with subsample size $\tilde{n} = \lfloor 0.632n \rfloor$ and selection probability cutoff $\pi_{thr} = 0.6$ (the R package defaults; see below). We do not place any explicit control on the expected number of false positives $\mathbb{E}[V]$ (i.e. we consider the full range of regularization parameters Λ). We avoid explicit false positive control as there will be no choice of upper bound that is optimal for all simulation settings considered. However, we assess sensitivity to the above choices in Section 3.1.

We use available R packages to implement the methods: `glmnet` for Lasso,

				Pairwise			Toeplitz
				s_0^B			s_0^B
p	p^B	$B = \frac{p}{p^B}$	s_0	1	2	5	2
500	100	5	10	✗	✓	✓	✓
			20	✗	✗	✓	✗
			40	✗	✗	✗	✗
1000	100	10	10	✓	✓	✓	✓
			20	✗	✓	✓	✓
			40	✗	✗	✓	✗
2000	100	20	10	✓	✓	✓	✓
			20	✓	✓	✓	✓
			40	✗	✓	✓	✓
4000	100	40	10	✓	✓	✓	✓
			20	✓	✓	✓	✓
			40	✓	✓	✓	✓
*	10	*	*	✓	✓	✓	✗

Table 2.2: Combinations of p, p^B, s_0 and s_0^B explored in the correlation designs. ✓ indicates that the combination is included and ✗ indicates that the combination is not included. For $p^B = 10$, * denotes all combinations of p and s_0. Note that for the correlation design, the s_0^B signals per block applies to the first s_0/s_0^B blocks only.

Elastic Net and Ridge Regression [Friedman et al., 2010]; `ncvreg` for SCAD [Breheny and Huang, 2011]; `flare` for Dantzig Selector [Li et al., 2015]; and `c060` for Stability Selection [Sill et al., 2014]. Variables are standardised and the response vector is centred. We run all methods on all simulation settings with the exception of Dantzig; for correlation designs, Dantzig is run only for $p = 500$ and $p = 1000$ due to its computational demands under multicollinearity for large p. For each simulation setting, we show results averaged across 64 simulated datasets (simulations are paralleled on 16 cores of each node in high performance computing cluster, and each core runs four simulations).

2.2.3 Performance metrics

We use metrics described in Section 1.7 to assess performance of a method. Specifically, we use pAUC to assess ranking and restrict to a maximum of 50 false positives (FPR $= \frac{50}{p-s_0}$). The pAUC calculation requires a score under which to rank variables j. For Ridge Regression, we rank by $s_j = |(\hat{\beta}_{cv})_j|$ ($\hat{\beta}_{cv}$ denotes coefficient estimate based on cross-validation) and for Stability Selection by $s_j = \max_{\lambda \in \Lambda} \hat{\Pi}_{\lambda,j}$. For the other methods (Lasso, Elastic Net, SCAD and Dantzig Selector), we could use $|(\hat{\beta}_{cv})_j|$ as for Ridge, but due to sparsity this would involve ranking many variables with $(\hat{\beta}_{cv})_j = 0$. We instead consider the set of estimated active sets $\{\hat{S}_\lambda : \lambda \in \Lambda\}$ where Λ is the set of candidate regularization parameters. We consider a variable to be more important the longer it remains in $\hat{S}_\lambda$ as λ increases and more sparsity is induced. This motivates defining ranking scores as: $s_j = \max\{\tilde{\lambda} \in \Lambda : j \in \hat{S}_\lambda \text{ for all } \lambda \leq \tilde{\lambda}, \lambda \in \Lambda\}$ or $s_j = 0$ if $j \notin \hat{S}_{\lambda_{\min}}$, where $\lambda_{\min} = \min\{\lambda \in \Lambda\}$. Although ranking with λ is not how the methods are designed, the reasoning behind this approach is that a variable is more likely to be important if it can survive under a higher level of penalty.

We use RMSE (1.34) to assess prediction with a test data of $n_{test} = 500$ samples. Stability Selection focuses on variable selection and we therefore do not include it in assessment of predictive performance.

We use TPR and PPV (1.35) to assess variable selection. Ridge Regression does not perform variable selection *per se* and is therefore excluded from the evaluation of variable selection.

2.3 Results using synthetic data

Due to the large number of simulation regimes, we focus below mainly on the key patterns. All performance data and plotting code are made available on GitHub, allowing specific scenarios to be investigated further.

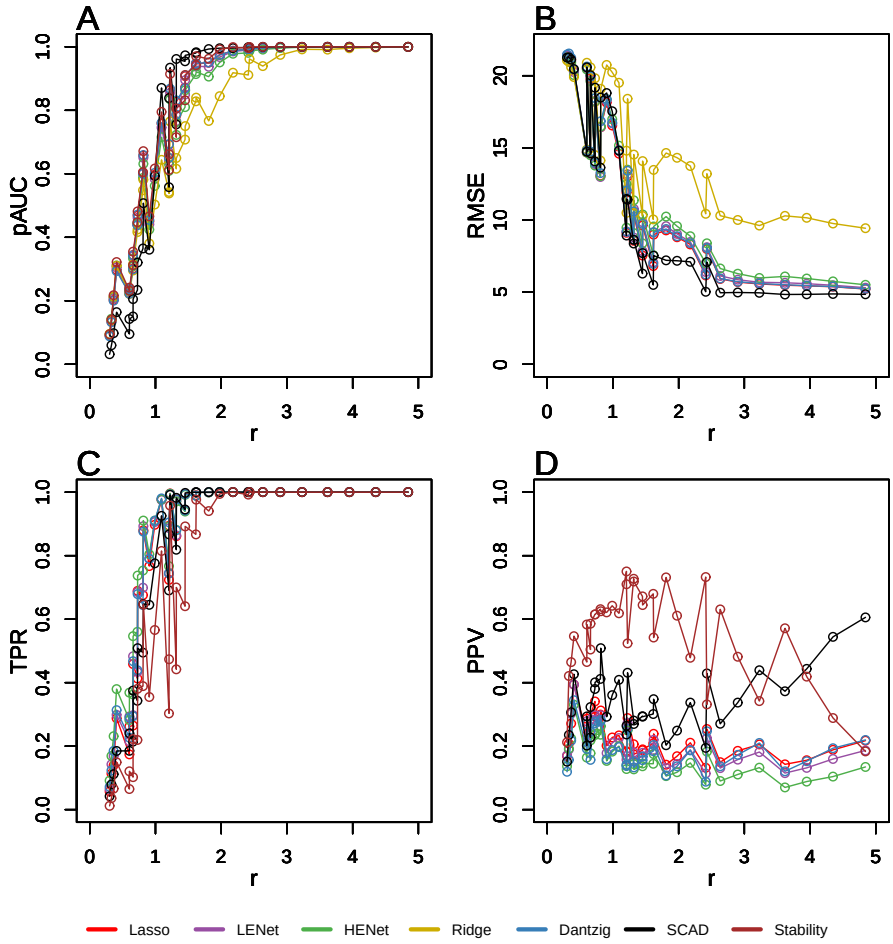


Figure 2.1: Ranking (A), prediction (B) and selection (C,D) performance versus the rescaled sample size $r = n/(s_0 log(p - s_0))$ for independence design scenarios with SNR=2. Line colour indicates method. Note that Stability Selection and Ridge Regression are not included in the assessment of prediction and selection performance respectively.

2.3.1 Independence design

2.3.1.1 Approximate guide to simulation setting difficulty

Recall that consistent variable selection for Lasso requires $s_0 log(p) = o(n)$ (see Section 1.4.2.2). This property motivates the rescaled sample size $r = n/(s_0 \log(p - s_0))$[see Wainwright, 2009]. Figure 2.1 shows the performance metrics versus the quantity r, for the independence design with SNR=2. Large (small) values of r can be interpreted as large (small) sample size relative to dimensionality and sparsity. We observe a clear overall trend of better pAUC (Fig. 2.1A) and TPR (Fig. 2.1C) for all methods as r increases, with performance leveling off for larger values of r. The trend is similar for RMSE as r increases, although with more local variation in performance (Fig. 2.1B). The behavior of PPV is method-dependent and the overall trend is non-monotonic as r increases (Fig. 2.1D). Performance with varying r was qualitatively similar for other SNR values and we also observed an expected trend of deteriorating performance with decreasing SNR (see Figs. 2.A1 and 2.A2 in Appendix 2.A for SNR=1 and SNR=4 respectively). Therefore, although the motivation for r lies in asymptotic theory for variable selection, r and SNR together serve as a useful approximate guide to the difficulty of each simulation scenario for all three tasks (selection, ranking and prediction). We make use of this characterization below.

2.3.1.2 Key observations

We first present an overview of the results before discussing the individual performance metrics in more detail below. Figures 2.2, 2.3 and 2.4 show ranking, selection and prediction performance respectively for a subset of independence design scenarios, while Figures 2.A4 and 2.A5 plot the performance of pairs of methods against each other across all scenarios.

Key observations for the independence design are:

I1 *No overall winner; large differences.* For all metrics, there is no one method that consistently performs best across all or the majority of the independence design scenarios. Moreover, relative differences in method performance can be large in some scenarios. For example, in Figure 2.2B, for p=4,000, there

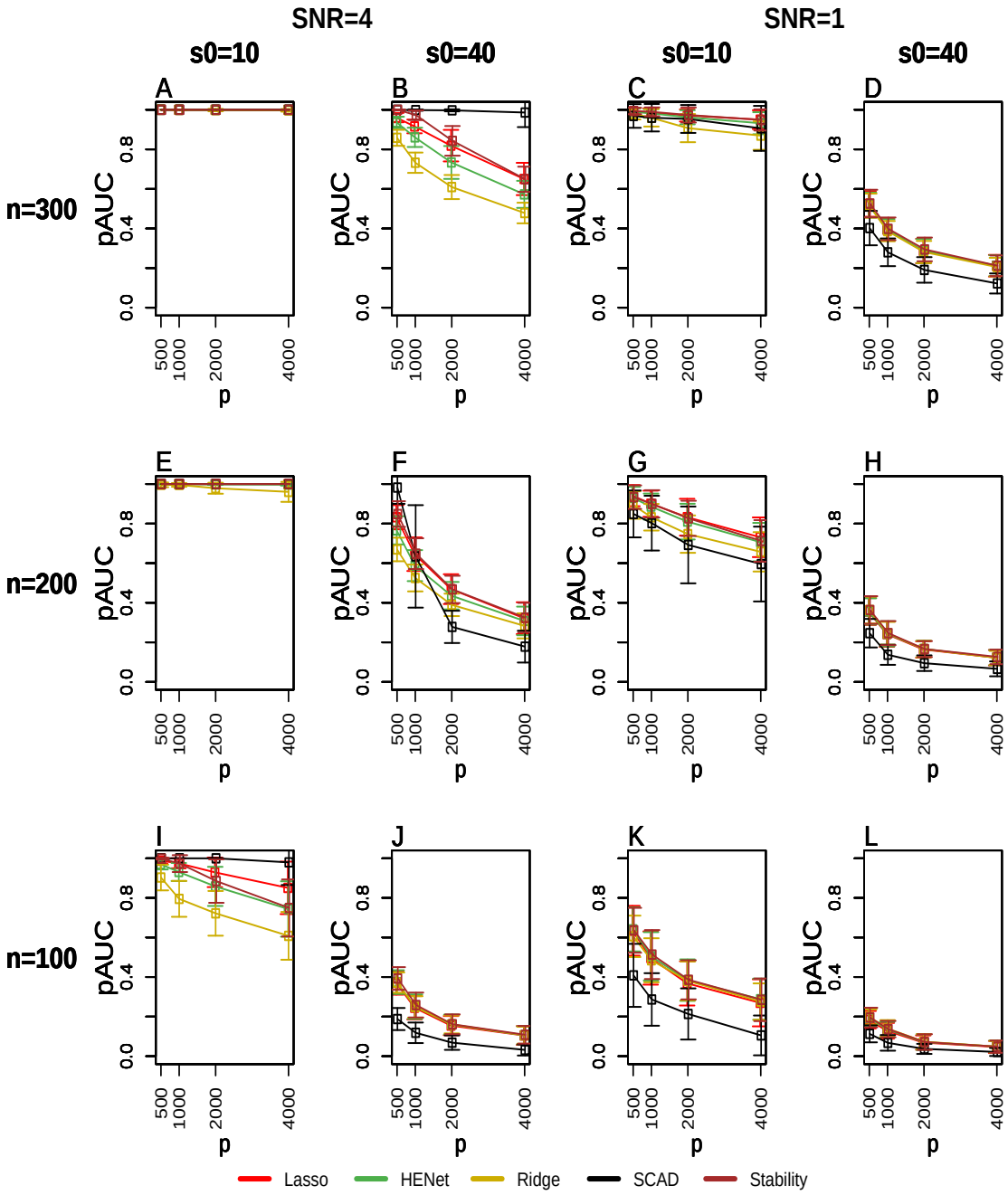


Figure 2.2: Ranking performance (pAUC) versus p for a subset of independence design scenarios. Each panel represents a different combination of n, s_0 and SNR. Line colour indicates method. Note that Dantzig Selector has similar performance to Lasso and is not shown; LENet is also not shown here, nor in subsequent figures, as its performance is invariably between that of Lasso and HENet.

is a percentage relative decrease in pAUC of 52% between the methods with the highest and lowest scores (SCAD and Ridge Regression respectively). Across all 108 independence design scenarios, the median percentage relative decrease is 25% for pAUC, 15% for RMSE, 30% for TPR and 70% for PPV.

I2 *SCAD transition.* The performance of SCAD relative to other methods varies substantially across scenarios. SCAD can offer the best performance in "easier" scenarios, but does not retain this advantage as scenario difficulty increases. In particular, for ranking, SCAD undergoes a transition from best to worst performing method (see e.g. black line in Fig. 2.2F). A similar transition is also seen for prediction (see below).

I3 *Stability Selection typically best for PPV; trade-off between PPV and TPR.* For selection, Stability Selection and SCAD typically outperform other methods in terms of PPV, with large gains in some scenarios and with Stability Selection offering the best performance except in "easy" scenarios. However, Stability Selection and SCAD often suffer from an inferior TPR (see, for example, circle symbols in Fig. 2.3B). In general, there is a trade-off between TPR and PPV. The reason for this trade-off is that higher TPR is typically associated with larger model, which includes more irrelevant variables.

I4 *Lasso performs well.* Under the independence design Lasso is competitive in the majority of scenarios for all metrics except PPV. SCAD can outperform Lasso in "easy" scenarios, but Lasso can be considered as a "safe" option because its relative performance across scenarios is less variable than for SCAD.

We will see below that these key observations largely continue to hold in the correlation designs.

As expected, there is no benefit of using an l_2 penalty under the independence design; mostly Lasso outperforms or is competitive with Elastic Net, which itself outperforms or is competitive with Ridge (see red, green and yellow lines in Fig. 2.1 and see also Figs. 2.A4 and 2.A5). An exception is for the selection metric TPR (see below). We completely exclude LENet from our presentation below due to its performance being invariably between that of Lasso and HENet.

The Dantzig Selector mostly performed similarly to Lasso (Figs. 2.A4 and 2.A5), in line with theory [e.g. Efron et al., 2007; Meinshausen et al., 2007]. However, Dantzig Selector is more computationally expensive than Lasso [Meinshausen et al., 2007]. For example, when $(n, p, s_0) = (100, 500, 10)$ and SNR=1, Dantzig Selector takes around 1,500 seconds to compute the whole solution path, while Lasso takes less than one second. In the interest of brevity, we only include Dantzig Selector below when its performance differs to Lasso.

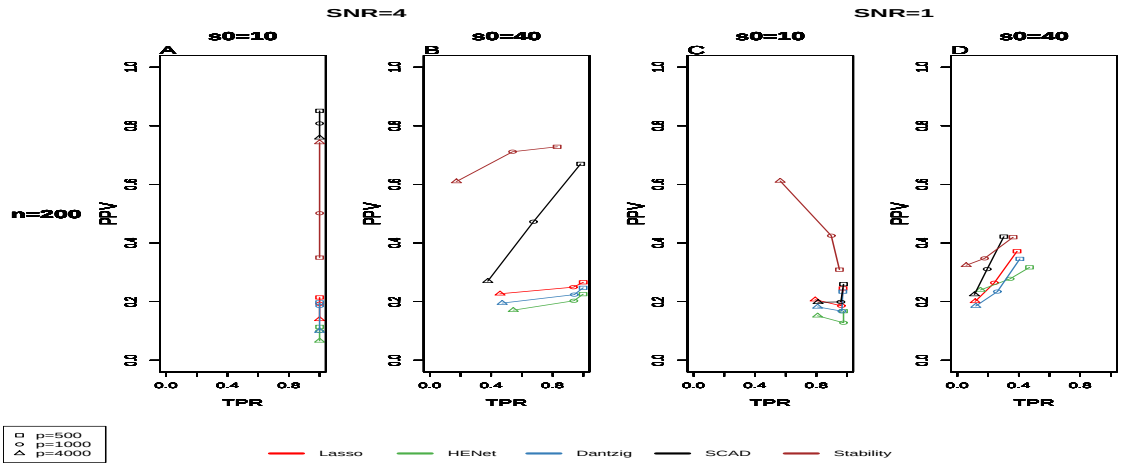


Figure 2.3: Selection performance for a subset of independence design scenarios with $n = 200$. Each panel represents a different combination of s_0 and SNR, and plots PPV against TPR for three values of p. Line colour indicates method and symbols indicate the value of p. Results for $n = 100$ and $n = 300$ are shown in Figure 2.A3.

2.3.1.3 Results by performance metric

Ranking. Ranking performance deteriorates for all methods as SNR or r decreases, but SCAD retains its good performance for longest and achieves the best performance in some "easier" scenarios (e.g. Fig. 2.2B, black line). However, SCAD transitions from best- to worst-performing method with an unfavourable change in n, p, s_0 or SNR (see Fig. 2.2F for such a transition with increasing p; see also Key Observation I2). The reason for this transition may be due to SCAD's oracle property, such that in 'easier' scenarios it achieves accurate coefficient estimation and variable selection; however, the model assumptions are sensitive to

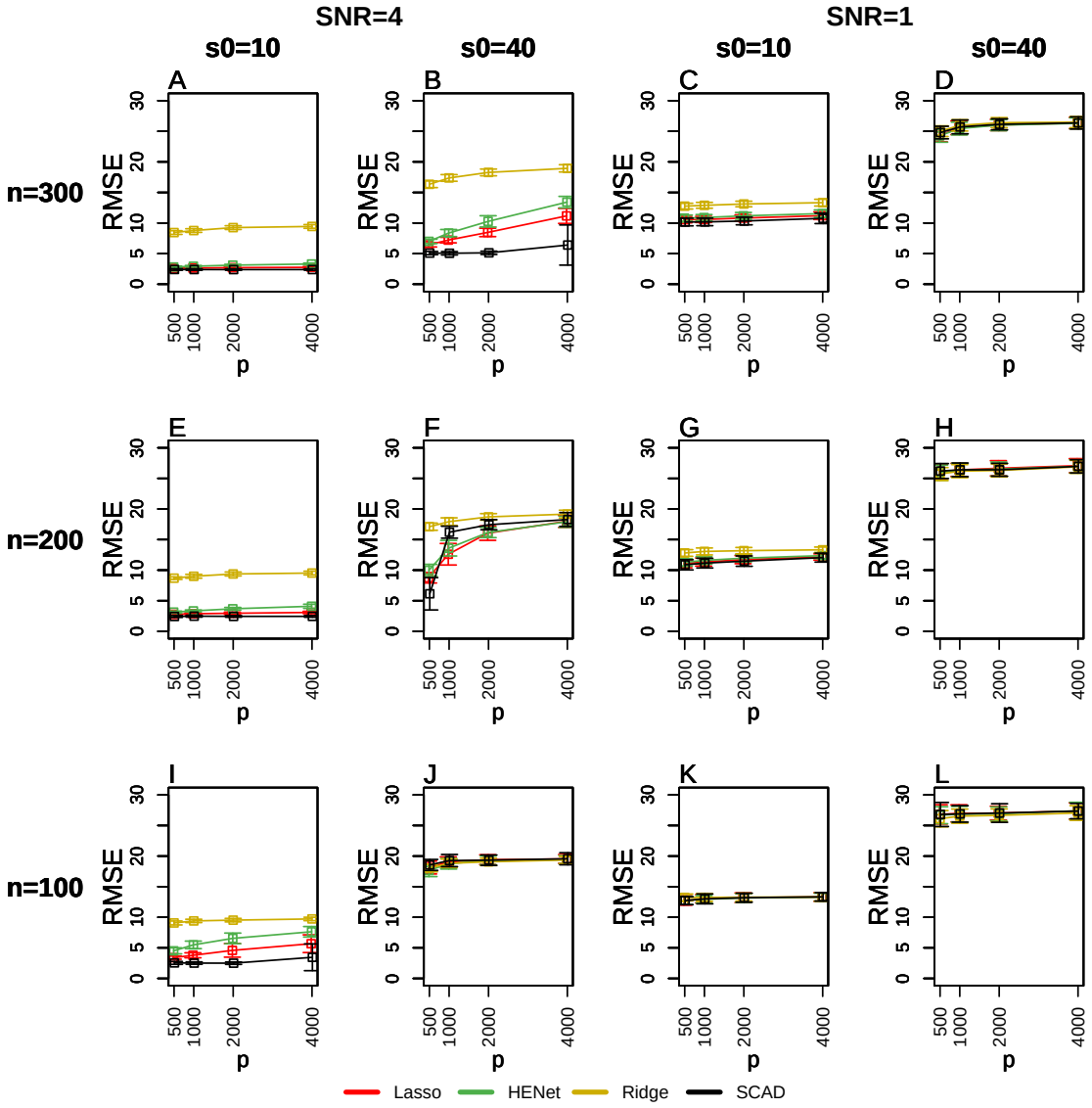


Figure 2.4: Prediction performance (RMSE) versus p for a subset of independence design scenarios. Each panel represents a different combination of n, s_0 and SNR. Line colour indicates method. Note that Dantzig Selector has similar performance to Lasso and is not shown, and prediction performance is not assessed for Stability Selection.

data properties, so its performance is more negatively affected than other methods in 'harder' scenarios. HENet and Ridge Regression fail to outperform Lasso in any scenario. Moreover, for some intermediate values of r, an l_2 penalty can even be detrimental for ranking with the worst performance provided by Ridge Regression (see e.g. yellow lines in Fig. 2.2B and I). Stability Selection has a very similar performance to Lasso (Fig. 2.A4). Thus, Lasso is competitive across all scenarios, except for those "easier" scenarios where SCAD performs best.

Prediction. Relative performance for prediction is broadly similar to that for ranking (contrast Fig. 2.2 with Fig. 2.4). In this sparse, independence setting we see again that an l_2 penalty offers no benefits, with Ridge performing substantially worse than all other methods in many scenarios. SCAD again shows transition behavior as difficulty increases, but it is never worse than Ridge, not even in "harder" scenarios.

Selection. All methods achieve optimal TPR when r and SNR are sufficiently large, but can at the same time have substantial differences in terms of PPV (see e.g. Fig. 2.3A; range of PPVs$\approx$0.1$-$0.8). SCAD offers the best PPV in these "easier" scenarios, while Stability Selection typically outperforms Lasso and HENet. Note that the inferior performance of Stability Selection relative to SCAD in the "easiest" scenarios could at least in part be due to the lack of false positive control in the implementation used here.

In scenarios where TPR is sub-optimal (small-to-moderate values of r or small SNR), the relative performance of two methods typically follows the rule: if method A has a higher TPR than method B, then method A will have a lower PPV (see e.g. triangles in Fig. 2.3B). For the majority of these scenarios, Stability Selection has the highest PPV and lowest TPR, and SCAD performs similar to or better than Lasso and HENet in terms of PPV, but similar or worse in terms of TPR (see e.g. Figures 2.3B-D and 2.A5; performance differences are greater for larger SNR).

Across the majority of scenarios, Lasso has small gains in PPV (of at most 0.1) over HENet and Dantzig Selector has PPV similar to or slightly worse than Lasso (see e.g. red, green and blue lines in Fig. 2.3B). Again, the converse relationships are true for TPR. Lasso, HENet and Dantzig fail to obtain PPV higher than 0.4

across all scenarios, contrasting with a maximum PPV greater than 0.8 for SCAD or Stability Selection. However, they are competitive in terms of TPR.

2.3.2 Pairwise correlation design

2.3.2.1 Key observations

We again provide an overview of results and then discuss in more detail below.

Comparison with the independence design for fixed correlation designs. For simplicity, we initially focus on two pairwise correlation designs and compare performance with the independence design for varying r and SNR. Both designs have two signals per block (s_0^B=2) and intra-block pairwise correlation of ρ=0.7, but one has block size p^B=10 and the other, p^B=100. Figures 2.5 and 2.6 show the impact of correlation on method performance, relative to the independence design, for all values of r (i.e. p, s_0 and n) and SNR=2 (see Figs. 2.A6-2.A9 for SNR=1 and SNR=4). Figures 2.7, 2.8 and 2.9 show the relative performance of methods in the two pairwise correlation designs for ranking, prediction and selection respectively (see also Figures 2.A10-2.A13).

We have the following key observations:

C1 *Effect of correlation is method- and scenario-specific.* Broadly speaking, correlated variables have a negative effect on performance, but in some scenarios there are clear positive effects. Benefits from correlation typically occur when r is small and are more salient for ranking and selection when block size is small (p^B=10; contrast Figures 2.5A and C with Figures 2.6A and C), or for prediction when block size is large (p^B=100, contrast Figure 2.5B with Figure 2.6B). SCAD tends to be the most negatively affected by correlation, while Ridge Regression and Elastic Net can, in some scenarios, be the least negatively affected or most positively affected (see e.g. black and yellow symbols in Fig. 2.6B).

C2 *Benefits of l_2 regularization in some "hard" correlated scenarios.* In line with theory, we find that an l_2 penalty confers small gains in ranking and prediction over Lasso in some correlated scenarios, resulting in Ridge or HENet

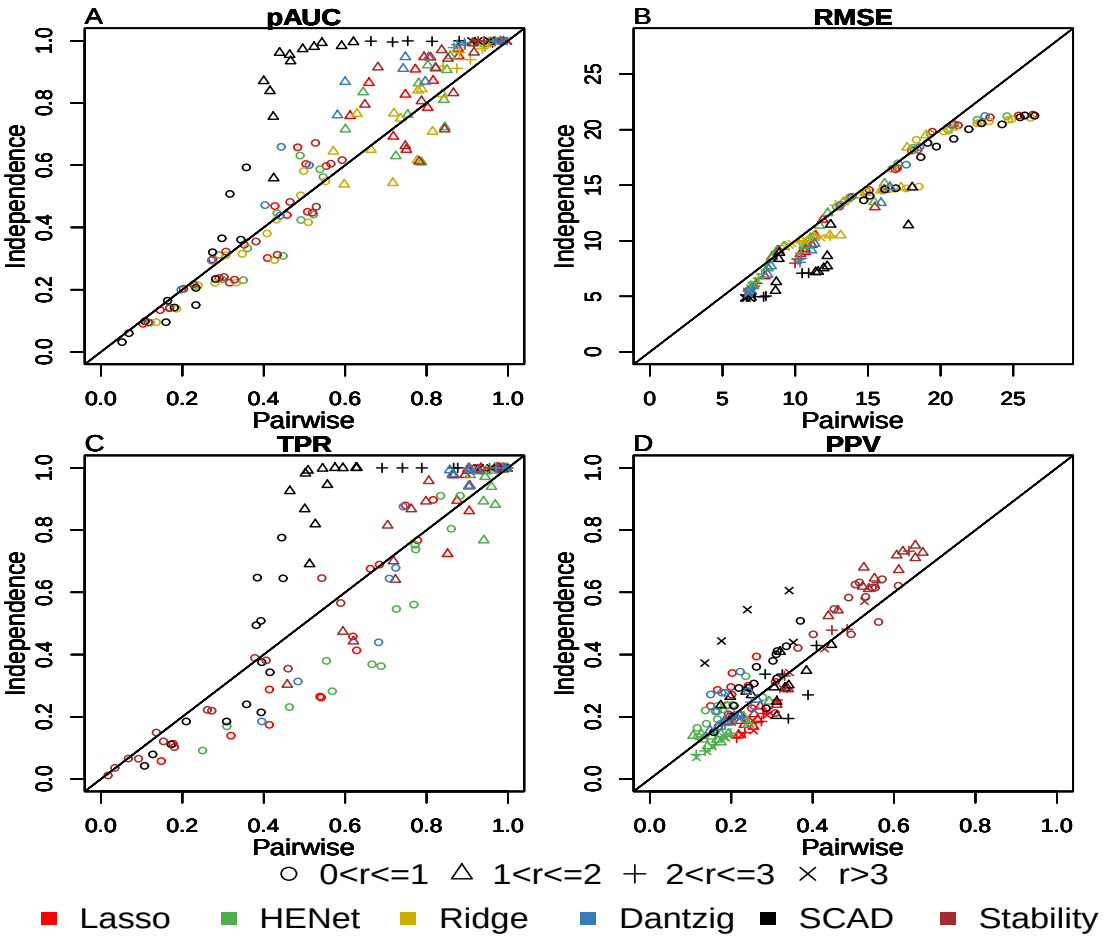


Figure 2.5: Influence of correlation on ranking (A), prediction (B) and selection (C,D) performance, relative to the independence design. Performance in the independence design is plotted against performance in the pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 10$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol). Points further from the diagonal represent a stronger influence of correlation. Results shown are for SNR=2 (see Figs. 2.A6 and 2.A7 for SNR=1 and SNR=4).

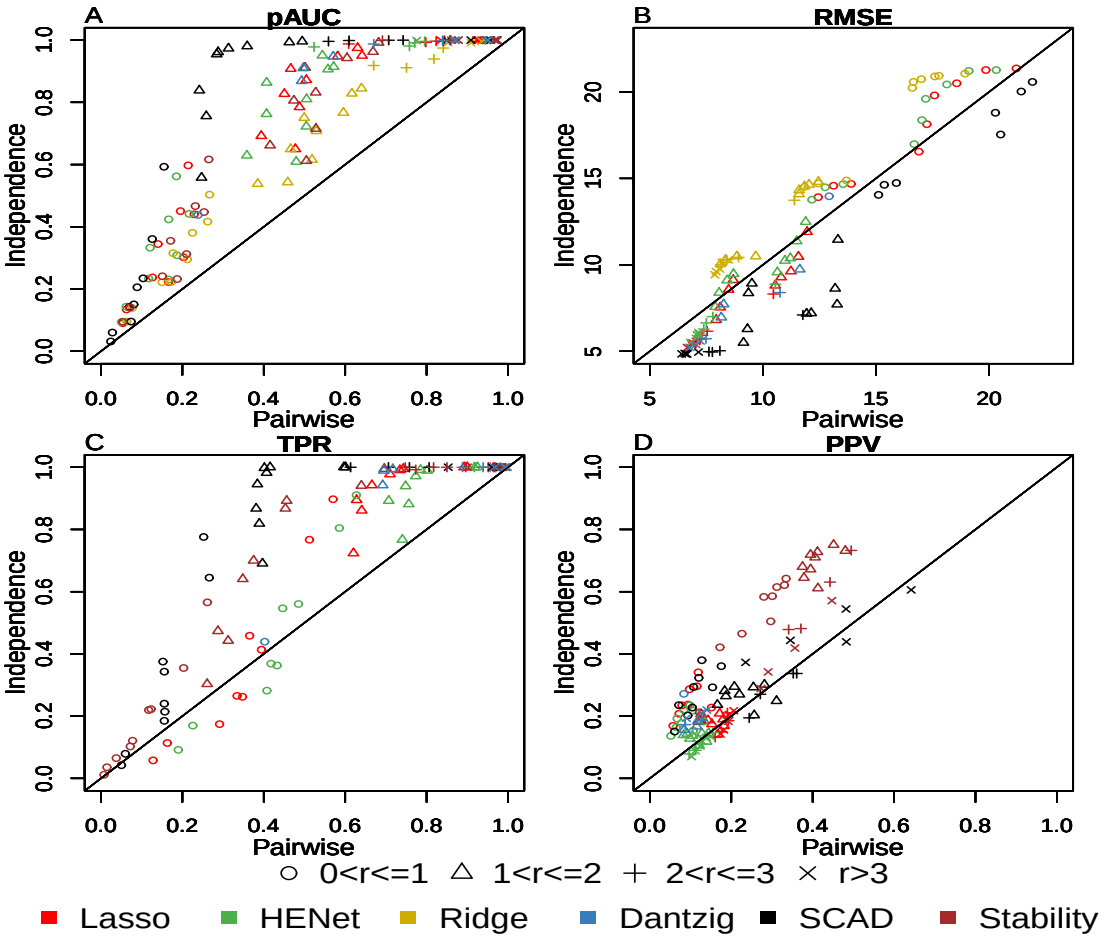


Figure 2.6: As Figure 2.5, except the pairwise correlation design has $p^B = 100$ instead of $p^B = 10$. Results shown are for SNR=2 (see Figs. 2.A8 and 2.A9 for SNR=1 and SNR=4).

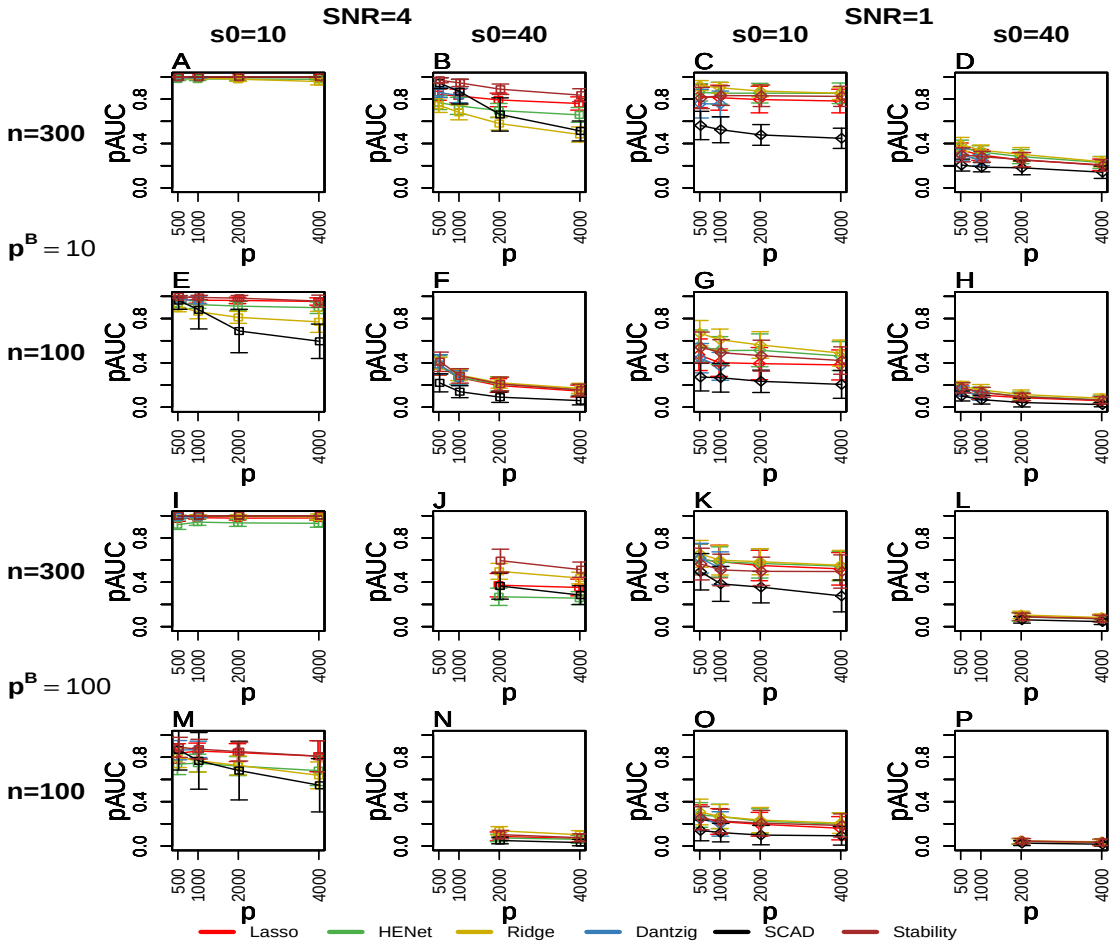


Figure 2.7: Ranking performance (pAUC) versus p for the pairwise correlation designs with $\rho = 0.7$, $s_0^B = 2$ and either $p^B = 10$ (top two rows) or $p^B = 100$ (bottom two rows). Each panel represents a different combination of n, s_0, SNR and p^B. Line colour indicates method. Note that some data points are missing when $p^B = 100$ and $s_0 = 40$ because the corresponding scenarios violate the necessary constraint $s_0^B \geq s_0/B$ (see Table 2.2), and Dantzig performance is only available for $p = 500$ and $p = 1000$ due to computational constraints.

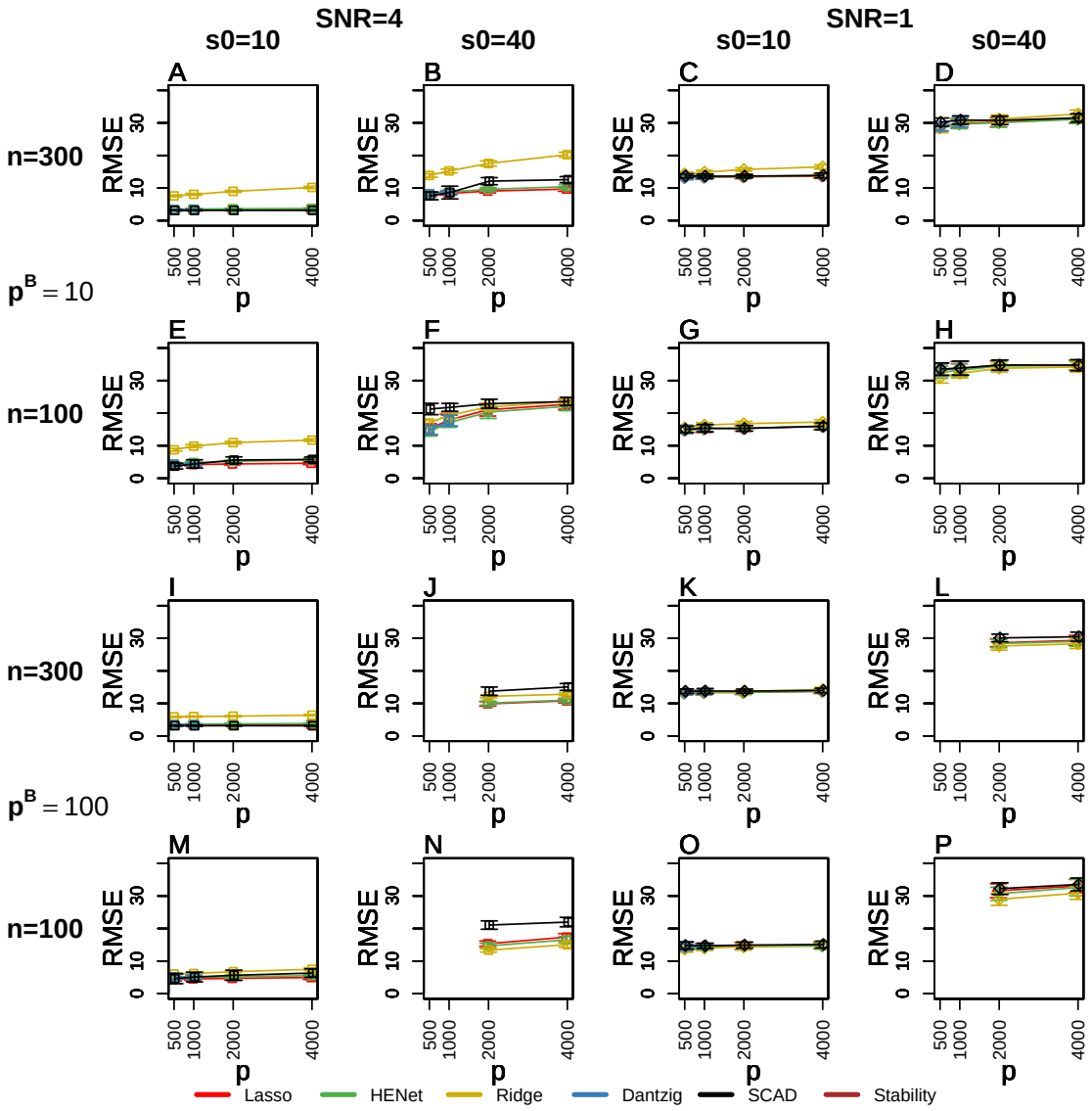


Figure 2.8: Prediction performance (RMSE) versus p for the pairwise correlation designs with $\rho = 0.7$, $s_0^B = 2$ and either $p^B = 10$ (top two rows) or $p^B = 100$ (bottom two rows). Each panel represents a different combination of n, s_0, SNR and p^B. Line colour indicates method. Note that some data points are missing when $p^B = 100$ and $s_0 = 40$ because the corresponding scenarios violate the necessary constraint $s_0^B \geq s_0/B$ (see Table 2.2), and Dantzig performance is only available for $p = 500$ and $p = 1000$ due to computational constraints.

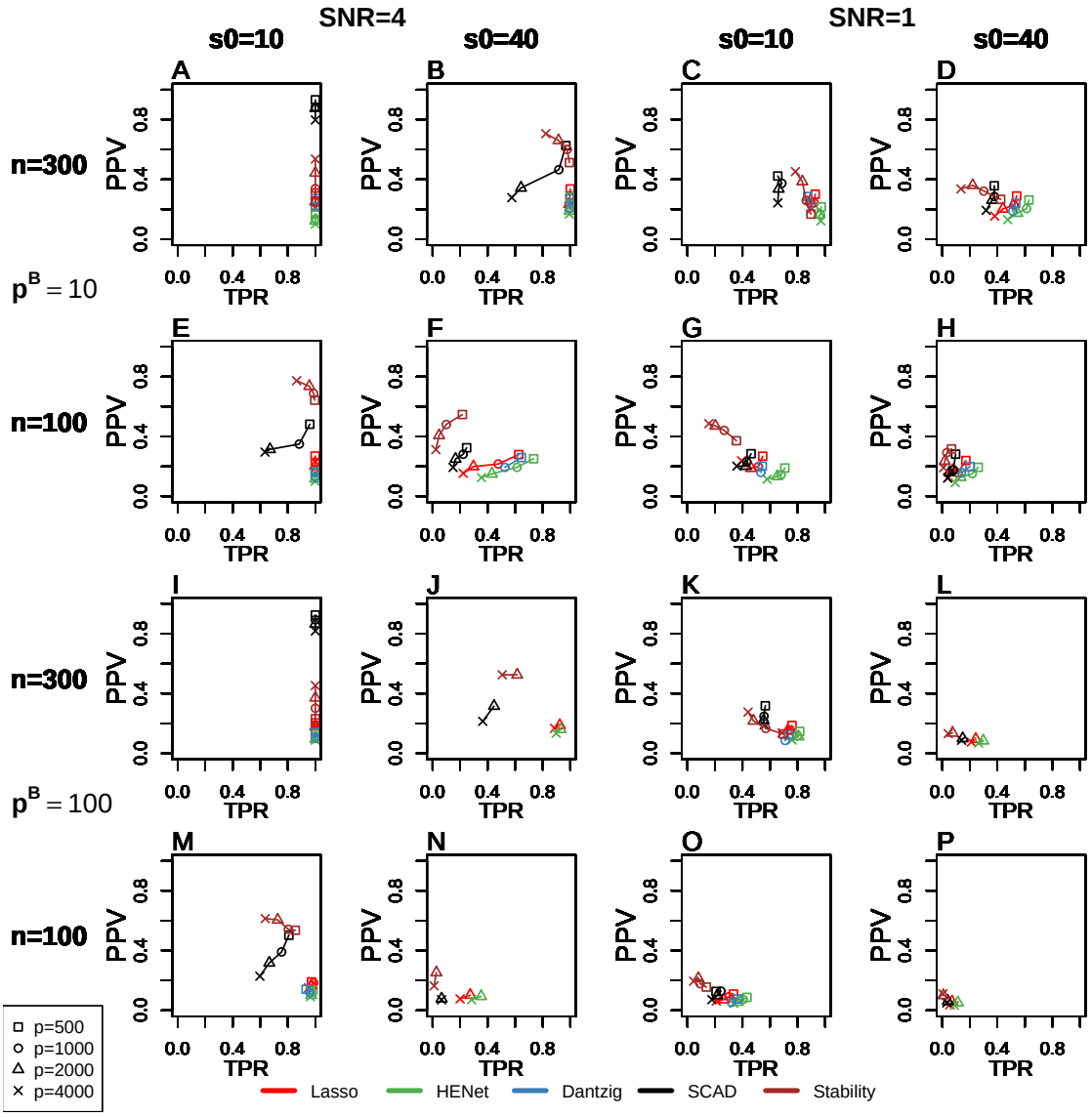


Figure 2.9: Selection performance for the pairwise correlation designs with $\rho = 0.7$, $s_0^B = 2$ and either $p^B = 10$ (top two rows) or $p^B = 100$ (bottom two rows). Each panel represents a different combination of n, s_0, SNR and p^B, and plots PPV against TPR for four values of p. Line colour indicates method and symbols indicate the value of p. Note that some data points are missing when $p^B = 100$ and $s_0 = 40$ because the corresponding scenarios violate the necessary constraint $s_0^B \geq s_0/B$ (see Table 2.2), and Dantzig performance is only available for $p = 500$ and $p = 1000$ due to computational constraints.

performing the best. We typically observe these gains in "harder" scenarios with small r or SNR (see e.g. yellow, green and red lines in Figure 2.7G).

Key Observations I1-I4 ("No overall winner", "SCAD transition", "Stability Selection best for PPV" and "Lasso performs well") from the independence design also still hold in these two correlation designs, with relative performance between methods remaining broadly the same.

Influence of correlation design parameters. We now turn our attention to the influence of the pairwise correlation design parameters p^B (block size), s_0^B (number of signals per block) and ρ (intra-block pairwise correlation). To aid presentation of results, we fix (n, p, s_0)=$(300, 4000, 40)$ or $(300, 1000, 10)$ which give r=0.91 ("hard") or 4.35 ("easy") respectively. Ranking performance is presented in Figure 2.10 and analogous results for prediction can be found in Figure 2.11. Selection performance is shown in Figure 2.12.

Key observations are:

C3 *Influence of correlation parameters can be positive or negative.* For ranking and selection, the influence of correlation ρ and number of signals per block s_0^B depends on scenario difficulty. In "easy" scenarios (with sufficiently large r or SNR), performance typically declines with increasing ρ or s_0^B (see e.g. Figs. 2.10O, 2.10P, 2.12O and 2.12P) . However, in "harder" scenarios (with small r or SNR), performance can improve with increasing ρ or s_0^B, particularly for HENet and Ridge Regression when block size p^B is small (see e.g. Figs. 2.10E, 2.10F, 2.12E and 2.12F; see also Key Observation C1). An increase in block size p^B typically has a negative effect or little effect on ranking and selection performance. Predictive performance generally worsens (or remains relatively stable) with an increasing number of signals per block s_0^B (for example, contrast Fig. 2.11E with Fig. 2.11F), while the influence of increasing correlation ρ has a strong dependence on block size p^B and s_0^B (see below for details). An increase in p^B typically has a positive effect or little effect on prediction.

C4 *Benefits of l_2 regularization depend on correlation parameters.* The gains in ranking performance from an l_2 penalty over Lasso in some "hard" scenarios

(see Key Observation C2) can be substantial, particularly when block size p^B is small, and correlation strength ρ and number of signals per block s_0^B are large (e.g. Fig. 2.10F). Similar benefits of an l_2 penalty are observed for selection with the TPR metric, which is in line with Elastic Net enjoying the grouping effect property for highly correlated variables (e.g. Fig. 2.12F). For prediction, advantages of an l_2 penalty observed in certain scenarios are very small (see below for details).

C5 *Stability Selection is competitive for ranking and selection (PPV).* Stability Selection is competitive for ranking across the majority of scenarios and performs best for some large SNR scenarios (see Figs. 2.10C and 2.10D). As in the independence design, it also typically performs best in terms of PPV (Key Observation I3; Fig. 2.12A-D).

C6 *Lasso remains competitive in many correlated scenarios.* Lasso remains competitive across many of the correlations designs for all metrics except PPV. Ridge Regression or HENet can offer substantive gains over Lasso for ranking and selection (TPR), particularly for small blocks that contain highly correlated variables, several of which are signals (see Key Observation C4). No such gains are observed for prediction. SCAD continues to perform best in a small number of scenarios that are "easy" (large r or SNR) and have weak correlation (small ρ and s_0^B).

Key observations I1 ("No overall winner)", I2 ("SCAD transition") and I3 ("Stability Selection best for PPV") from the independence design still hold across the correlation designs and Key Observation C6 is an updated version of Key Observation I4 ("Lasso performs well").

The reason why correlation can have positive effect on method performance may be that effects of positively correlated variables are in the same direction, and exaggerate one another, so their collective effect is stronger than when variables are independent from each other.

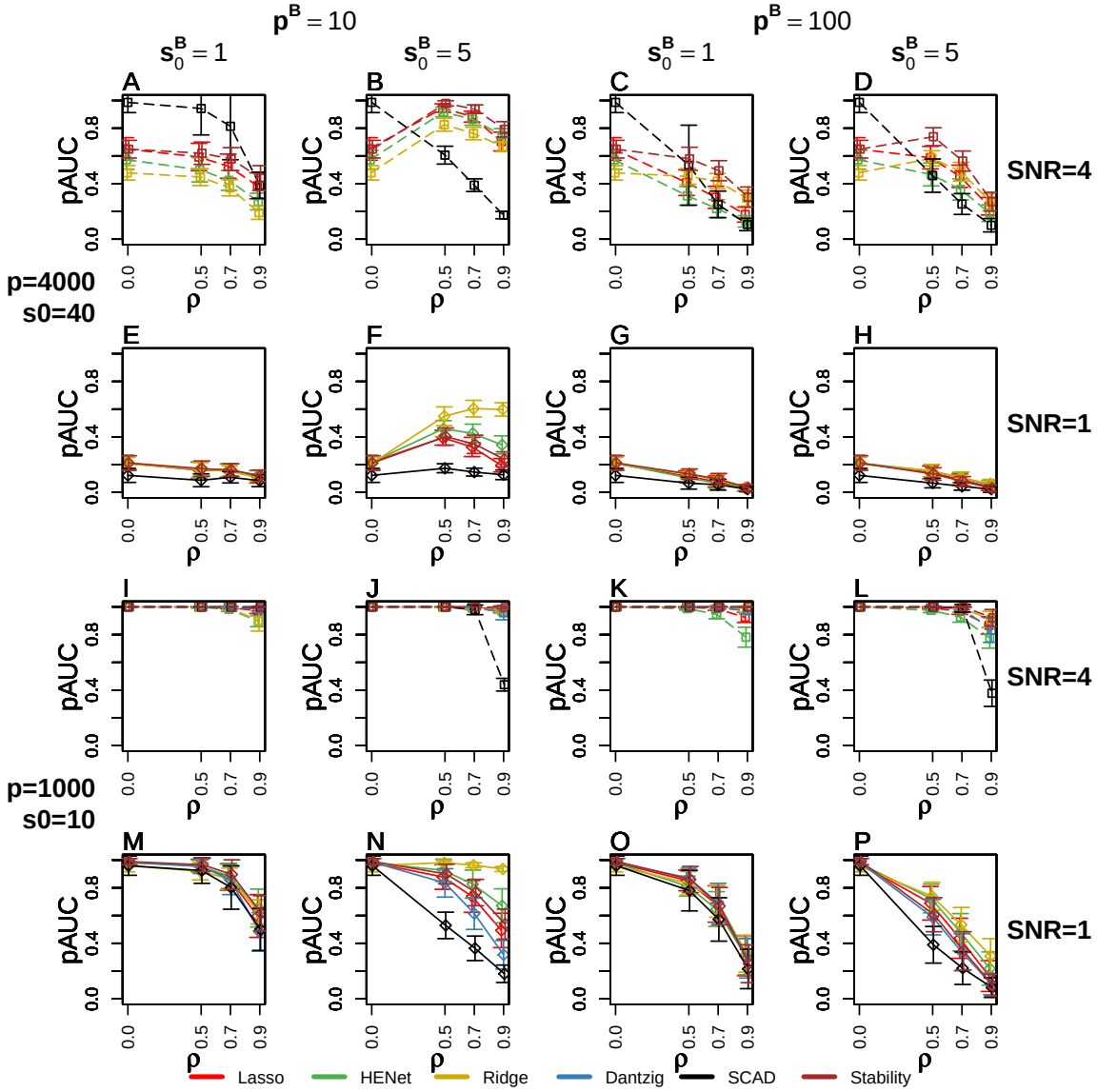


Figure 2.10: Ranking performance (pAUC) versus ρ for a subset of pairwise correlation designs. Each row represents a different combination of p, s_0 and SNR, while each column represents a different combination of p^B and s_0^B. All results shown are for $n = 300$. The top two rows have $(n, p, s_0) = (300, 4000, 40)$, giving $r = 0.91$, and the bottom two rows have $(n, p, s_0) = (300, 1000, 10)$, giving $r = 4.35$. For comparison, results for the corresponding independence design scenarios are also shown ($\rho = 0$; these data points are identical across the panels in each row). Line colour indicates method.

2.3.2.2 Results by performance metric

Ranking. We saw in Key Observations C1 and C3 above that there is a method- and scenario-specific influence of correlation (also see explanation in Section 2.3.2.1). The most salient benefits from correlation are seen for HENet and Ridge in "hard" scenarios with small SNR or r when block size p^B is small and blocks consist of highly correlated variables of which several are active (i.e. large ρ and s_0^B). For example, when SNR=1, $r = 0.91$, $p^B = 10$ and $s_0^B = 5$ (Fig. 2.10F), all methods benefit from correlation, but HENet and Ridge have the largest improvements over the independence design. The gains from Ridge can be substantial with an increase in pAUC of 0.39 when $\rho = 0.9$. Note that improvements over the independence design are not typically monotonically increasing with ρ; the largest gains are often seen for moderate correlation strength (Figs. 2.10B and 2.10F).

This positive influence of correlation in "hard" scenarios with small, highly correlated blocks containing several signals gives rise to an improved ranking performance from an l_2 penalty over Lasso (Key Observations C2 and C4). Taking the same example as above (Fig. 2.10F), Ridge substantially outperforms all other methods when $\rho = 0.9$, with an improvement in pAUC of 0.25 over the second best method, HENet. HENet itself also improves over Lasso with a difference in pAUC of 0.14. So the magnitude of the gains over Lasso is linked to the strength of the l_2 penalty. Substantial improvements from an l_2 penalty over Lasso are not seen in the corresponding larger block size scenario ($p^B = 100$ with $s_0^B = 5$, SNR=1 and $r = 0.91$; Figure 2.10H), suggesting that the proportion of signals per block is important (see also Section 3.3).

SCAD again displays its characteristic transition behavior with increasing r or SNR (see e.g. Fig. 2.7), but due to it typically being the most negatively affected by correlation (Key Observation C1), the number of scenarios where SCAD performs best is reduced. SCAD's sensitivity to correlation also results in a transition with increasing ρ or s_0^B when SNR = 4; SCAD can perform best in scenarios with weak correlation and only one signal per block, but performs worst when correlation is strong and there are many signals per block (see e.g. black lines in Figs. 2.10A and 2.10B).

Stability Selection is competitive across the majority of correlation design sce-

narios and can perform the best in some SNR = 4 scenarios (see e.g. brown lines in Figs. 2.10C and 2.10D; see also Key Observation C5). The exceptions where it is not competitive are the scenarios described above where Ridge or SCAD are the best performers. It is also worth noting that Lasso can have modest gains over Dantzig Selector, in particular for larger s_0^B (Fig. 2.10N).

Prediction. The impact of increasing correlation strength ρ on predictive performance depends on block size p^B and number of signals per block s_0^B. The most salient improvements as ρ increases are observed for large p^B and small s_0^B, while the most notable declines occur for small p^B and larger s_0^B. The reason why prediction performance favors larger p^B and smaller s_0^B may be that in those scenarios, variables tend to have similar predictive power, and predictive performance is less sensitive to the variables selected.

For example, in Figure 2.11C where $p^B = 100$ and $s_0^B = 1$ in a large SNR, small r scenario, Ridge has a 57% reduction in RMSE relative to the independence design when $\rho = 0.9$. On the other hand, in Figure 2.11B where we have instead $p^B = 10$ and $s_0^B = 5$, Ridge has a 56% increase in RMSE. In general, Ridge is affected the most as correlation parameters change. This influence of ρ, taken together with possible positive effects of increasing p^B and negative effects of increasing s_0^B (Key Observation C3), means that correlation design scenarios with large, highly correlated blocks with signals spread across many blocks (i.e. large p^B and ρ, small s_0^B) are most favourable for prediction. These favourable correlation design scenarios have a large number of variables that are associated with the response (and so are, in a sense, non-sparse due to the correlation, even though each block is sparse in terms of number of signals) and Ridge, which has a non-sparse solution, typically benefits the most (as seen by contrasting the yellow lines in Figs. 2.11B and 2.11C).

Ridge does not substantively outperform the other methods for prediction in any of the scenarios considered here, even the favourable correlation design scenarios described above that benefit Ridge the most. In such scenarios, Ridge can marginally outperform other methods when r is very small (Key Observation C1), but performance remains poor. For example, for the most "difficult" scenario in Figure 2.8P (where $n = 100, p = 4000, s_0 = 40, \text{SNR} = 1, p^B = 100, \rho = 0.7$ and $s_0^B = 2$), RMSE=33.05 and 30.87 for Lasso and Ridge respectively. Note that this

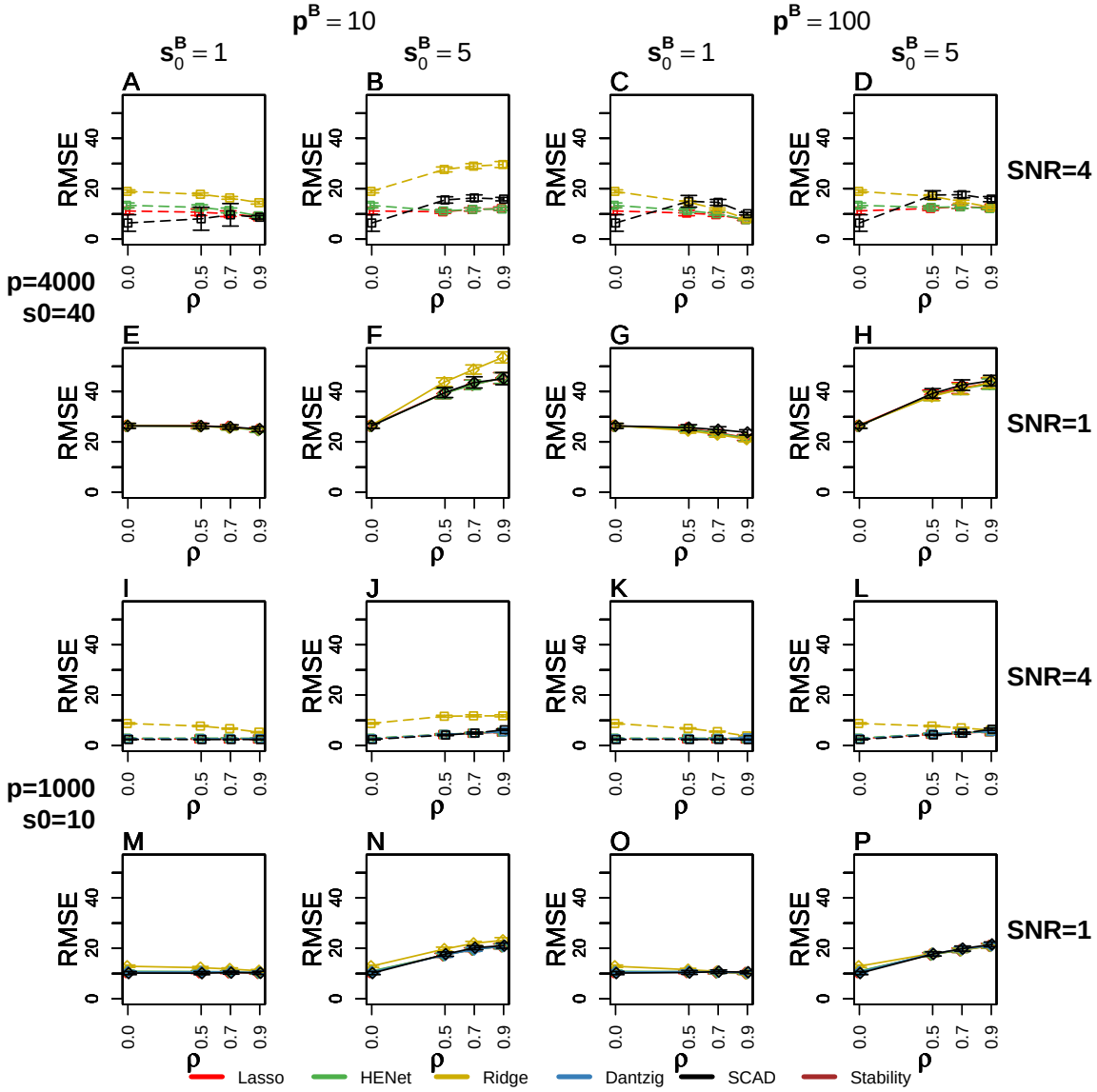


Figure 2.11: Prediction performance (RMSE) versus ρ for a subset of pairwise correlation designs. Each row represents a different combination of p, s_0 and SNR, while each column represents a different combination of p^B and s_0^B. All results shown are for $n = 300$. The top two rows have $(n, p, s_0) = (300, 4000, 40)$, giving $r = 0.91$, and the bottom two rows have $(n, p, s_0) = (300, 1000, 10)$, giving $r = 4.35$. For comparison, results for the corresponding independence design scenarios are also shown ($\rho = 0$; these data points are identical across the panels in each row). Line colour indicates method.

contrasts with ranking, where Ridge performed best for small p^B and large s_0^B.

SCAD again shows transition behavior, offering modest gains over other methods when r and SNR are large, and ρ and s_0^B are small (i.e. in "easy", weakly correlated scenarios), but becoming worse than Lasso, HENet and sometimes Ridge as scenario difficulty increases or correlation becomes stronger.

Selection. Results in the majority of the correlation designs mirror those seen in the independence design (see Key Observation I3). Stability Selection typically offers the largest PPV, outperforming SCAD, which in turn outperforms Lasso, which itself has small gains over HENet. Due to the trade-off between PPV and TPR, the opposite relation generally holds for TPR. As before, a notable exception is that SCAD can have the best PPV in "easy" scenarios with large r and this occurs across most correlation designs, as seen for $(n, p, s_0) = (300, 1000, 10)$ in Figure 2.12I-L.

Stability Selection and SCAD can be sensitive to correlation. For example, in the small r, large SNR scenario with large block size shown in Figure 2.12C and D, the substantial improvements in PPV provided by Stability Selection and SCAD in the independence design (see square symbols) are mostly or, in the case of SCAD, completely lost under stronger correlation ($\rho = 0.9$; see "+" symbols). SCAD also loses competitiveness in terms of TPR. Despite this sensitivity to correlation, Stability Selection still typically performs best except in "easy" scenarios.

Lasso is typically reasonably competitive in terms of TPR, but as outlined in Key Observations C1-C4 and similar to ranking, an l_2 penalty gives substantial improvements in TPR over Lasso in "hard" scenarios with small, highly correlated blocks and a large proportion of signals per block. For example, in Figure 2.12F where SNR=1, $r = 0.91$, $p^B = 10$ and $s_0^B = 5$, HENet has a TPR of 0.56 when $\rho = 0.9$ (green "+" symbol) compared with 0.31 for Lasso (red "+" symbol). At the same time, PPV remains competitive at 0.22 for HENet and 0.23 for Lasso.

Note that Dantzig Selector, in the scenarios where results are available ($p = 1000$), can perform slightly worse than Lasso in terms of TPR and PPV (see e.g. blue and red lines in Figure 2.12P).

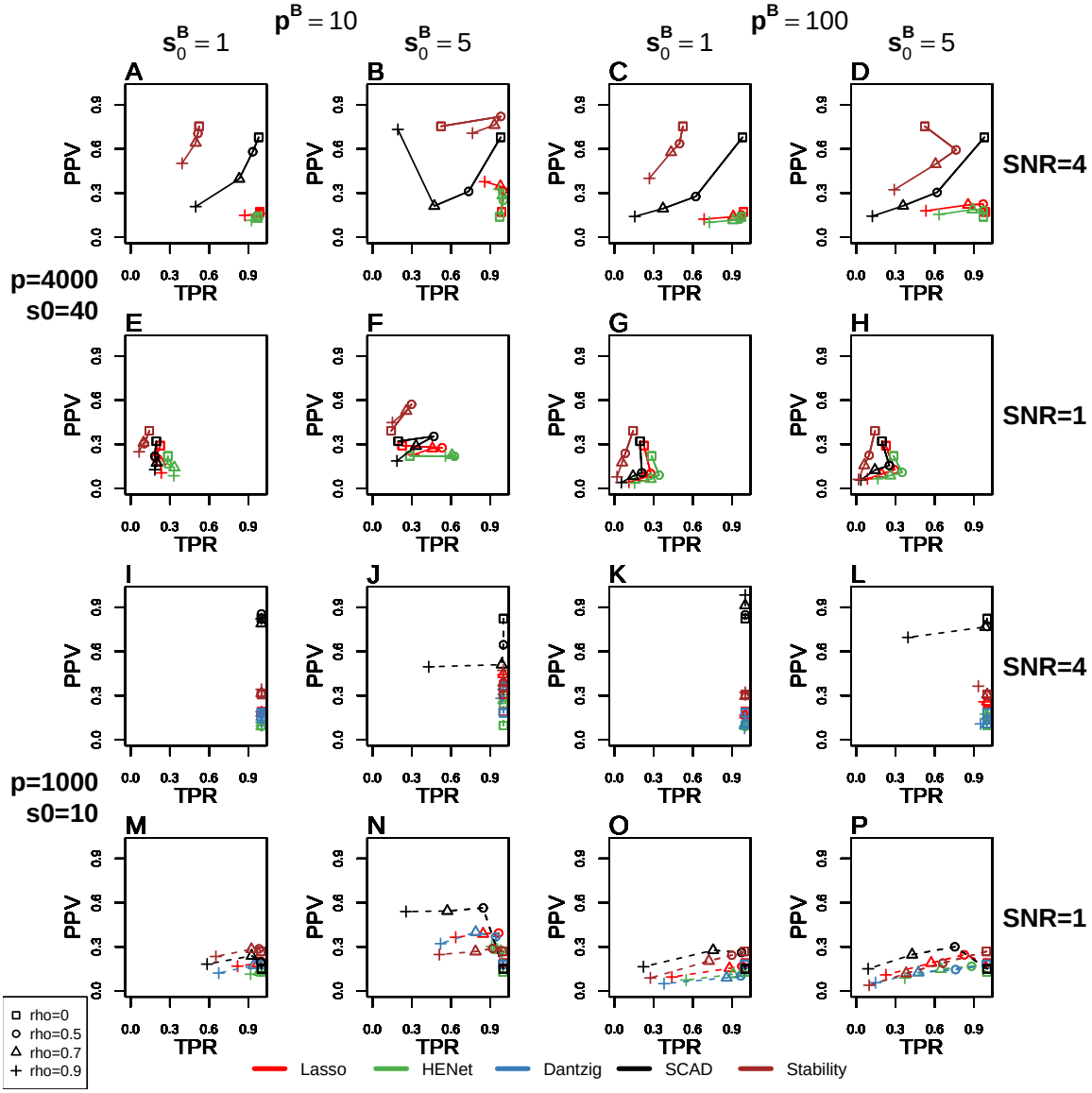


Figure 2.12: Selection performance for a subset of pairwise correlation design scenarios. Each panel plots PPV against TPR for four values of ρ, including $\rho = 0$ (independence design). Line colour indicates method and symbols indicate the value of ρ. Each row represents a different combination of p, s_0 and SNR, while each column represents a different combination of p^B and s_0^B. All results shown are for $n = 300$. The top two rows have $(n, p, s_0) = (300, 4000, 40)$, giving $r = 0.91$, and the bottom two rows have $(n, p, s_0) = (300, 1000, 10)$, giving $r = 4.35$.

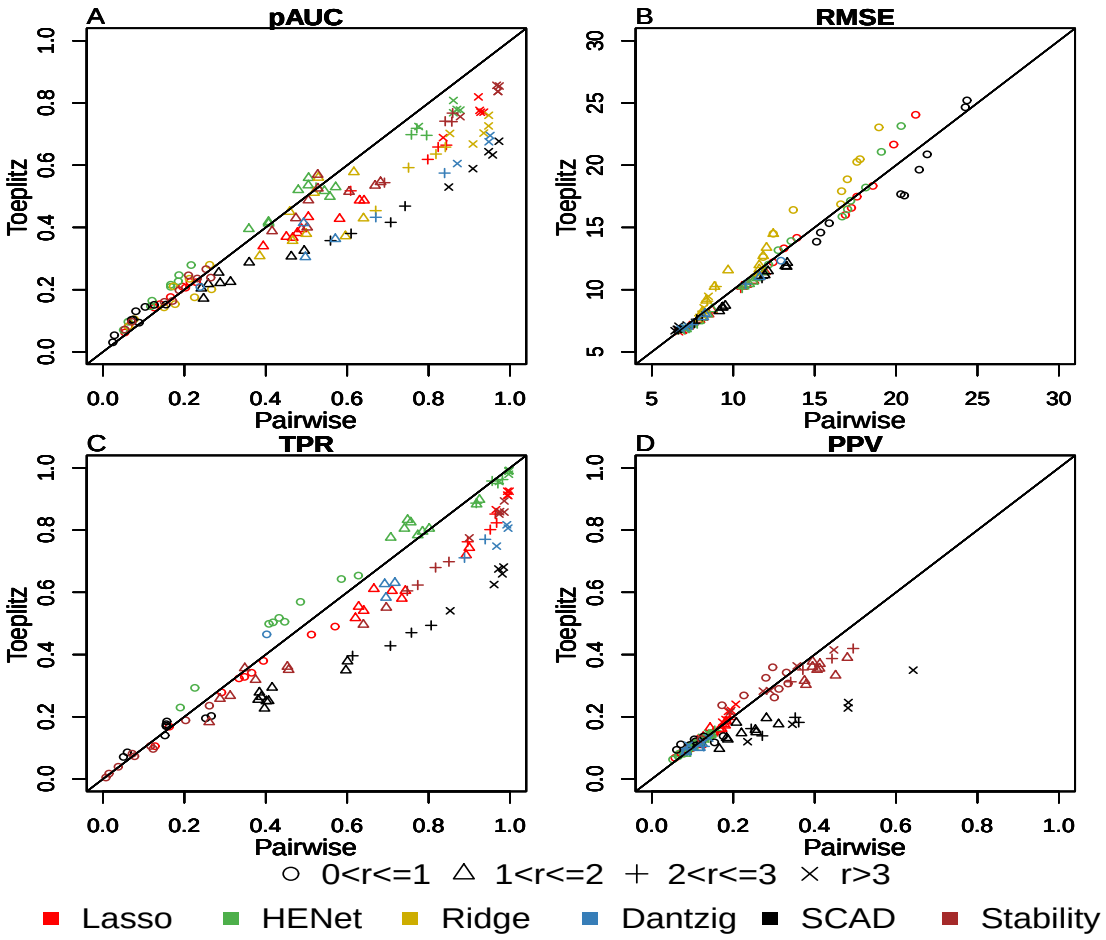


Figure 2.13: Comparison between Toeplitz correlation and pairwise correlation designs for ranking (A), prediction (B) and selection (C,D) performance. Performance in the Toeplitz correlation design (see Section 2.2.1) is plotted against performance in the corresponding pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 100$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol). Results shown are for SNR=2 (see Figs. 2.A14 and 2.A15 for SNR=1 and SNR=4).

2.3.3 Toeplitz correlation design

We now consider method performance in the Toeplitz correlation design with block size p^B=100 and number of signals per block s_0^B=2 where the two signals are correlated with $\rho \approx 0.7$ (see Section 2.2.1). Figure 2.13 compares performance in the Toeplitz design against that in the corresponding pairwise correlation design ($\rho = 0.7$) for SNR = 2 and all possible combinations of n, p and s_0 (see Figs. 2.A14 and 2.A15 for SNR=1 and SNR=4 respectively). Performance is typically similar for the two designs or worse in the Toeplitz design. For prediction, Ridge Regression is most negatively affected by Toeplitz correlation, while SCAD is most affected for the other metrics.

On the one hand, the pairwise correlation design could be considered more difficult than the Toeplitz design because the average correlation between signals and non-signals (within a block) is higher for pairwise than for Toeplitz (0.7 vs. 0.19). However, on the other hand, the Toeplitz design could be considered more difficult because there are several non-signals that are more strongly correlated with the signals than the signals are with each other; for the pairwise correlation design all signals and non-signals within a block are correlated with equal strength. The generally poorer performance observed for the Toeplitz design therefore suggests that having strongly correlated signals and non-signals is more detrimental than a higher average correlation.

Relative performance of methods in the Toeplitz design is generally consistent with that seen for the corresponding pairwise correlation design (contrast Figs. 2.A16 and 2.A17 with Figs. 2.A12 and 2.A13). For ranking, the impact of an l_2 penalty (relative to Lasso) is larger under the Toeplitz design; Ridge performs particularly well when SNR=1, but poorly when SNR=4 (see Fig. 2.A16).

2.4 Results using semi-synthetic data

To complement the purely simulated data above, we considered an example using real variables from The Cancer Genome Atlas (TCGA) study. We used gene expression data from TCGA ovarian cancer samples [TCGA, 2011], and specifically, we used the dataset provided in the Supplementary Appendix of Tucker et al.

[2014][1]. The dataset contained 594 samples and expression levels for 22,277 genes. The samples were a mixture of primary tumor (569), recurrent tumor (17) and normal tissue (8). We randomly subsampled the samples and genes to obtain a $n \times p$ design matrix $\mathbf{X} = [\mathbf{x_1}, \ldots, \mathbf{x_p}]$. A response vector $\mathbf{y}$ was then obtained using the linear model (1.1). Those samples not included in $\mathbf{X}$ were used as test data. Design matrix columns for both training and test data were centred and scaled to have zero mean and unit variance, and responses were centred.

Signals are allocated among the p variables to give either low or high correlation scenarios, using an approach similar to Bühlmann and Mandozzi [2014]. Specifically, for the low correlation scenarios, s_0 signals were randomly allocated among $\mathbf{x_1}, \ldots, \mathbf{x_p}$ and for the high correlation scenarios, we used the following procedure that mimics the main simulation set-up by forming correlated blocks: (i) form a block of p^B variables consisting of a randomly chosen variable $\tilde{\mathbf{x}}$ and the $p^B - 1$ variables that are most correlated with $\tilde{\mathbf{x}}$; (ii) allocate s_0^B signals to this block by designating $\tilde{\mathbf{x}}$ and the $s_0^B - 1$ variables that are most correlated to it as signals; (iii) repeat steps (i) and (ii), but remove from consideration any variables already allocated to a block, and continue repeating until all s_0 signals have been allocated.

We set $n = 100$, $p = 1000$, $s_0 = 10$ or 20 (giving $r = 1.45$ or 0.72 respectively), SNR $= 1, 2, 4$ or 8, and for the high correlation scenarios, $p^B = 10$ and $s_0^B = 5$. We generated 100 semi-synthetic datasets for each of the 16 scenarios. We applied the penalised regression methods to each scenario, but excluded the Dantzig Selector because the main simulations provided little evidence to prefer Dantzig over Lasso and Dantzig is computationally intensive. For Stability Selection, for ranking, there is no explicit false positive control and the full range of tuning parameters is considered when calculating selection probabilities. For selection, we chose $\tilde{V} = 10$, $\pi_{thr} = 0.6$, and subsample size $\tilde{n} = \lfloor 0.632n \rfloor$.

Ranking, prediction and selection performance are shown in Figure 2.14. Results are largely consistent with those from the main simulations; we highlight a few observations here. SCAD performs well in "easy" scenarios with large r and SNR, and weak correlation, but is less competitive otherwise (Key Observations I2 and C1; e.g., compare the solid black lines in Figs. 2.14A and 2.14E). An l_2 penalty

[1]The dataset is available at .http://bioinformatics.mdanderson.org/Supplements/ResidualDisease.

is useful for ranking when data is noisy or strong multicollinearity exists (Key Observation C4; e.g., compare yellow and red solid lines at SNR=1 and SNR=8 in Fig. 2.14E). SCAD and Stability Section are conservative and tend to have good false positive control but less power (Key Observation I3; see black and brown lines in Figs. 2.14C, D, G and H). Except for PPV, Lasso is overall competitive (Key Observations I4 and C6).

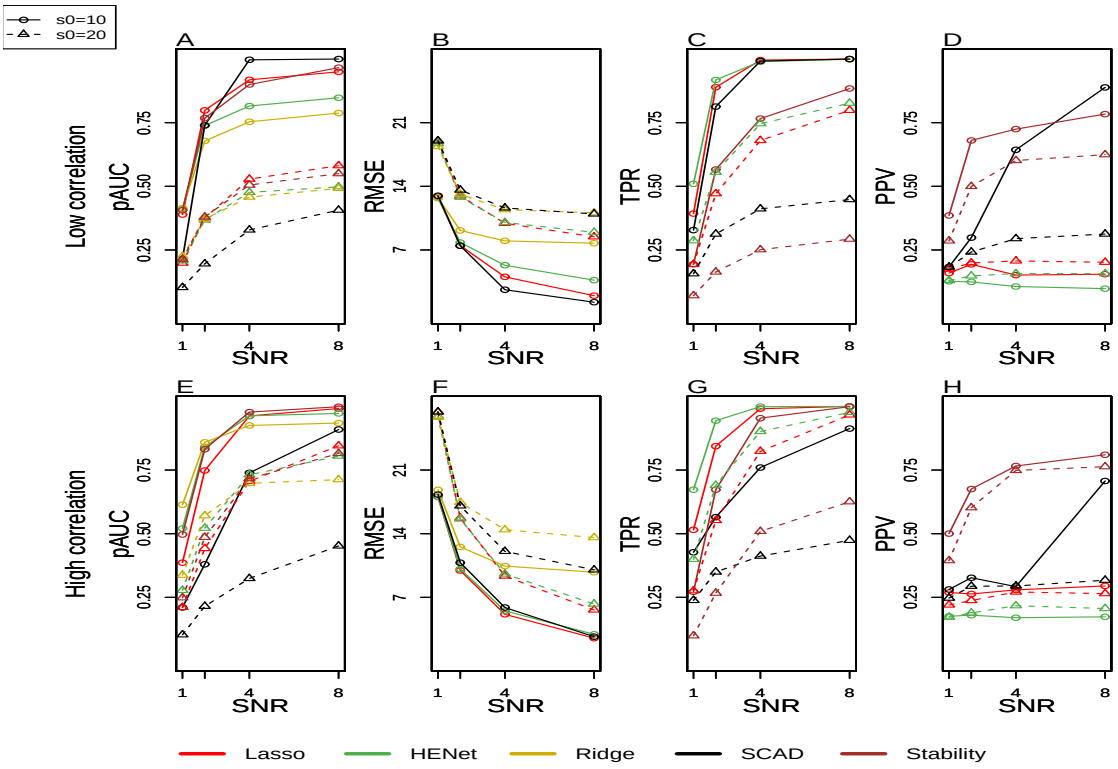


Figure 2.14: TCGA ovarian cancer expression data analysis. Ranking, prediction and selection performance (columns; left to right) versus SNR for $s_0 = 10$ ($r = 1.45$; solid lines) and $s_0 = 20$ ($r = 0.73$; dashed lines) in a low correlation setting (top row) and high correlation setting (bottom row). Line colour indicates method and results are averages over 100 semi-synthetic datasets for each scenario (see text for full details of semi-synthetic data generation).

2.5 Discussion

Our results complement theory by shedding light on the finite-sample relative performance of methods. Many of our results do align with available theory. For instance, SCAD is known to have nearly unbiased estimates for coefficients that are large (relative to noise), explaining why it tends to have better selection performance in "easy" scenarios. On the other hand, the Lasso and Elastic Net are biased, so good prediction requires more irrelevant variables to compensate. However, some conditions of theoretical results (asymptotic or finite-sample) can be hard to verify in practice, and the results do not directly provide insight into the performance of a method relative to others, making it difficult to pick a suitable method in any given finite-sample setting. Our results suggest that there is no one method which clearly dominates others in all scenarios, even in the relatively narrow set of possibilities considered here (e.g. we did not consider heavy tailed noise, non-sparsity, non-block-type covariance etc.). Relative performance depends on many factors, and also on the specific metric(s) of interest.

Nevertheless, with the above caveats, we can highlight some general observations. For data generated from sparse linear models: Lasso is relatively stable and outperforms Elastic Net and Ridge in the mild correlation designs; Elastic Net and Ridge only outperform Lasso in the most challenging, correlated design scenarios we considered here; SCAD is double-edged, dominating in "easier" scenarios but deteriorating rapidly when conditions become difficult; Stability Selection is good for false positive control and ranking; and Dantzig is usually similar or worse than Lasso. Ridge does particularly badly in many scenarios, but it is worth pointing out that most scenarios in this chapter were pro-Lasso (and unfriendly to Ridge) in the sense of being highly sparse, and with low overall correlation (across all variables). In many areas such as biomedicine, signals can be rather weak, and r can also be small. In such difficult settings, Ridge should perhaps receive more attention due to its robustness. SNR considered in this study is higher than in many scientific areas, which is a limitation, and is worth further investigation.

Finding the best level of penalty is especially challenging in high-dimensional data, and given different goals, the optimal penalty may vary. In line with known results, we saw that standard cross-validation often yielded overly large models for

Lasso, Elastic Net and the Dantzig Selector. An interesting alternative is proposed in Lim and Yu [2016], where cross-validation is based on an estimation stability metric. Compared to traditional cross-validation, this approach significantly reduces the false positive rate while slightly sacrificing the true positive rate, and achieves similar prediction performance but higher accuracy in parameter estimation. Stability is important in variable selection, in the sense that important variables should be selected even if small perturbations are added to data. This is desirable in biomarker detection such that researchers can be fairly confident on identified biomarkers before further biological investigation. Kirk et al. [2013] consider the balance between robustness and predictive performance in logistic regression problem, and conclude that combining the assessment of stability and predictive performance can help increase power while controlling false positives. In other words, a good choice of penalty level should take into account different performance goals and strike a balance.

Due to the comprehensive nature of our simulation study, we focus on summarizing the predominant trends and relationships across the scenarios. There will always be some scenarios which are exceptions to these summaries, but this in itself motivates the need for extensive simulation studies. If a simulation study has limited scope then the derived conclusions may not generalise beyond the few scenarios considered. So while such studies may be useful in exploring and understanding the properties of a method, they may have limited practical implications for an end-user. In contrast, a large-scale simulation study, such as the one presented here, offers some insight as to which method may be the most appropriate, depending on the properties of the data. Compared to Bühlmann and Mandozzi [2014], we see that the finer grid of simulation settings explored in our comparison study provides new insights, since we are able to elicit trends in performance as the properties of the data change.

A practical question of such large-scale comparison studies is how to present the results. On the one hand, high-level summaries extract general trends with respect to relative performance of different methods. However, they may be susceptible to exceptions and may not cover scenarios encountered in practice, so the conclusions are of less practical use. On the other hand, if too much detail is provided in the exposition, core information will get swamped in exceptions and specifics, and

conclusions can also be misleading, or hard to generalise to other scenarios. Thus striking a balance between the two is crucial for presenting the results, and in this chapter, observations are summarised to the degree that most typical types of scenarios are covered. We sacrifice certain details and exceptions such that conclusions can offer a guidance in practice, but we also keep those details and exceptions in mind when the comparative performance deviates significantly from expectation, in which case we investigate further to find out the driven factor, which is useful to fully understand the properties of different methods.

Code and data availability

All analysis was performed in R [R Core Team, 2017]. Scripts for generating the main simulation synthetic data sets, applying the regression methods, assessing performance and plotting results are available at `https://github.com/fw307/high_dimensional_regression_comparison`, together with performance metric data from the main simulation.

2.A Appendix: Additional figures for Chapter 2

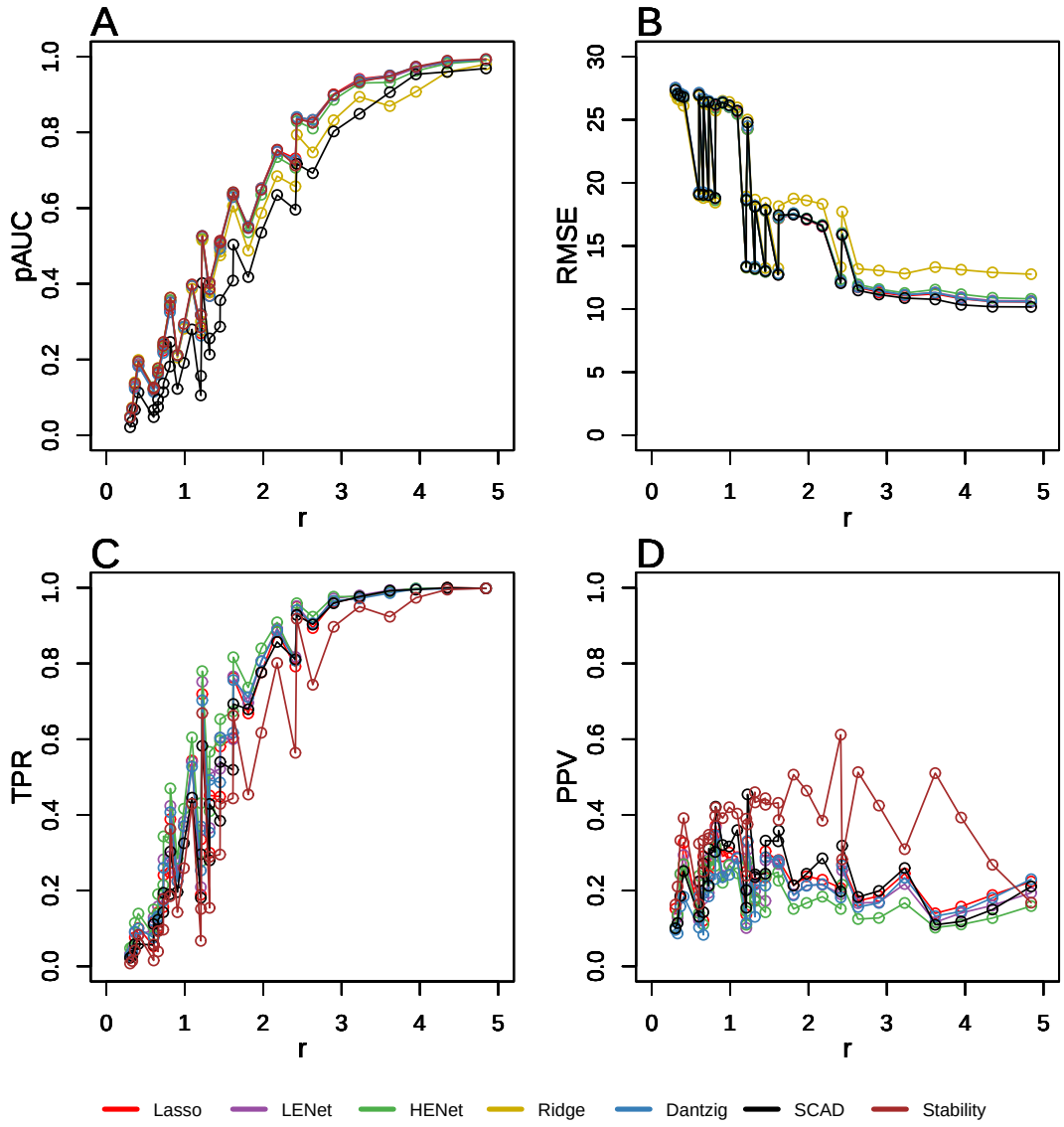


Figure 2.A1: Ranking (A), prediction (B) and selection (C,D) performance versus the rescaled sample size $r = n/(s_0 log(p - s_0))$ for independence design scenarios. As Figure 2.1, but with SNR=1 (instead of SNR=2).

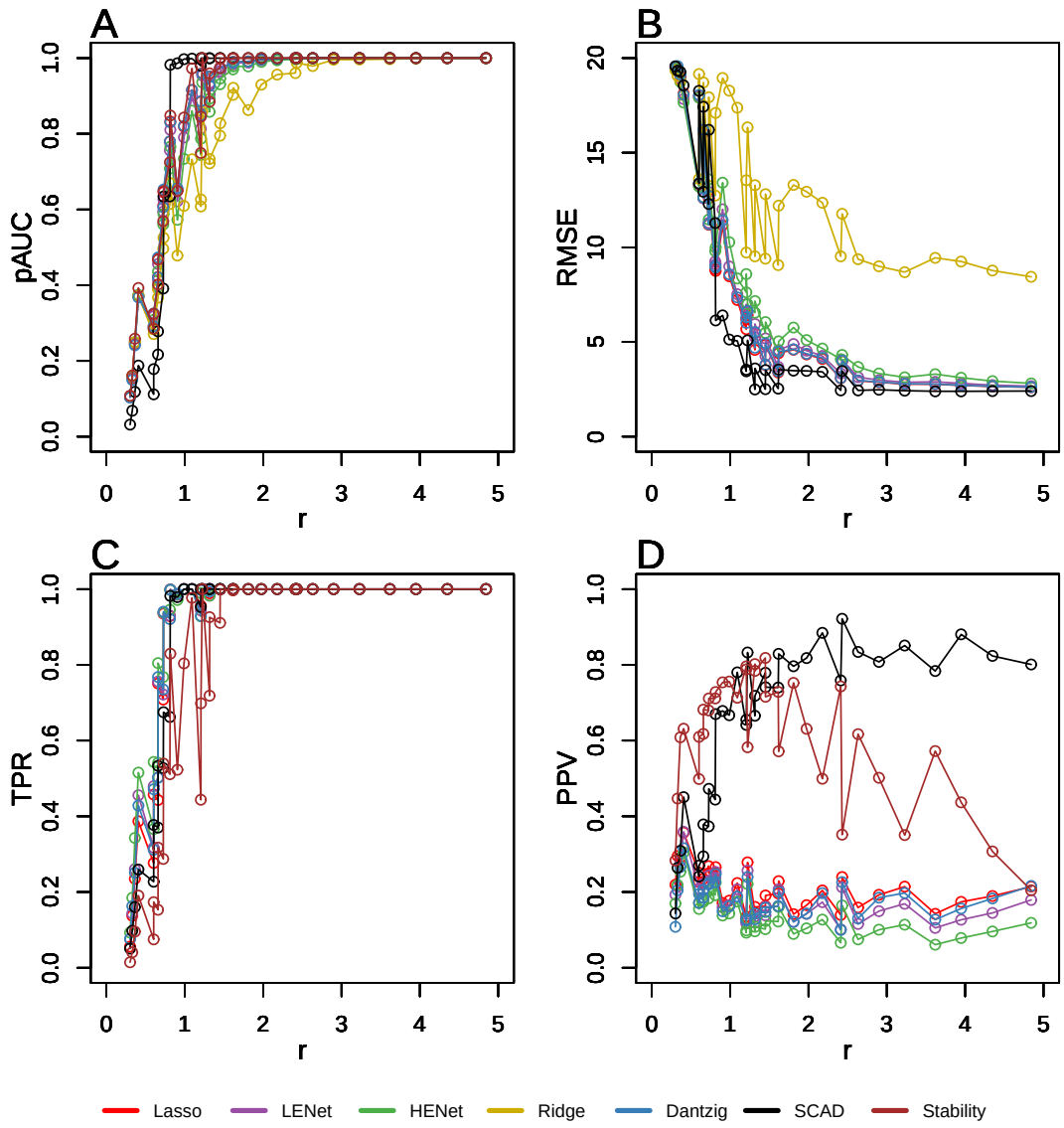


Figure 2.A2: Ranking (A), prediction (B) and selection (C,D) performance versus the rescaled sample size $r = n/(s_0 log(p - s_0))$ for independence design scenarios. As Figure 2.1, but with SNR=4 (instead of SNR=2).

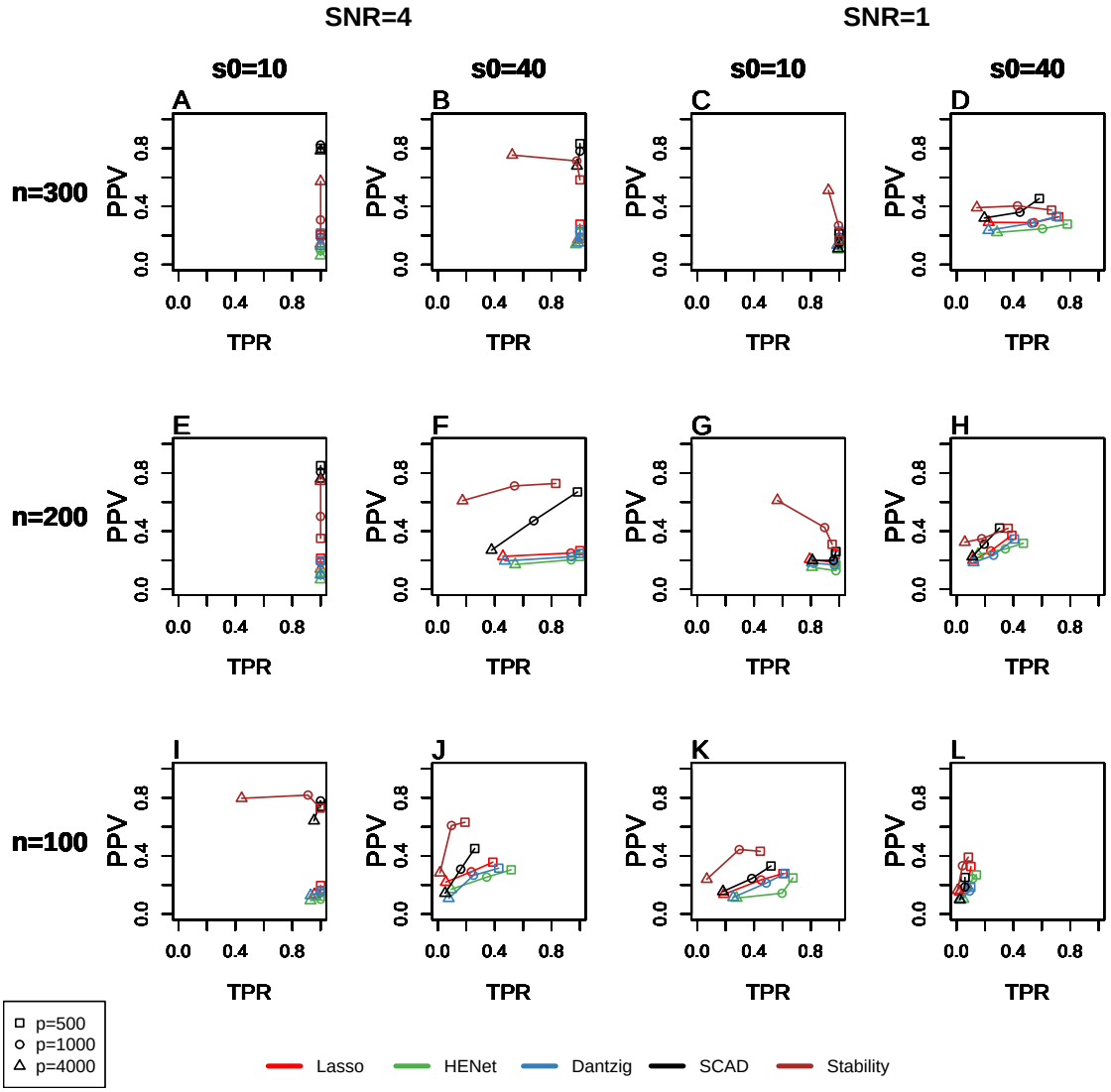


Figure 2.A3: Selection performance for a subset of independence design scenarios. As Figure 2.3, but with additional scenarios $n = 100$ and $n = 300$. Each panel represents a different combination of s_0 and SNR, and plots PPV against TPR for three values of p. Line colour indicates method and symbols indicate the value of p.

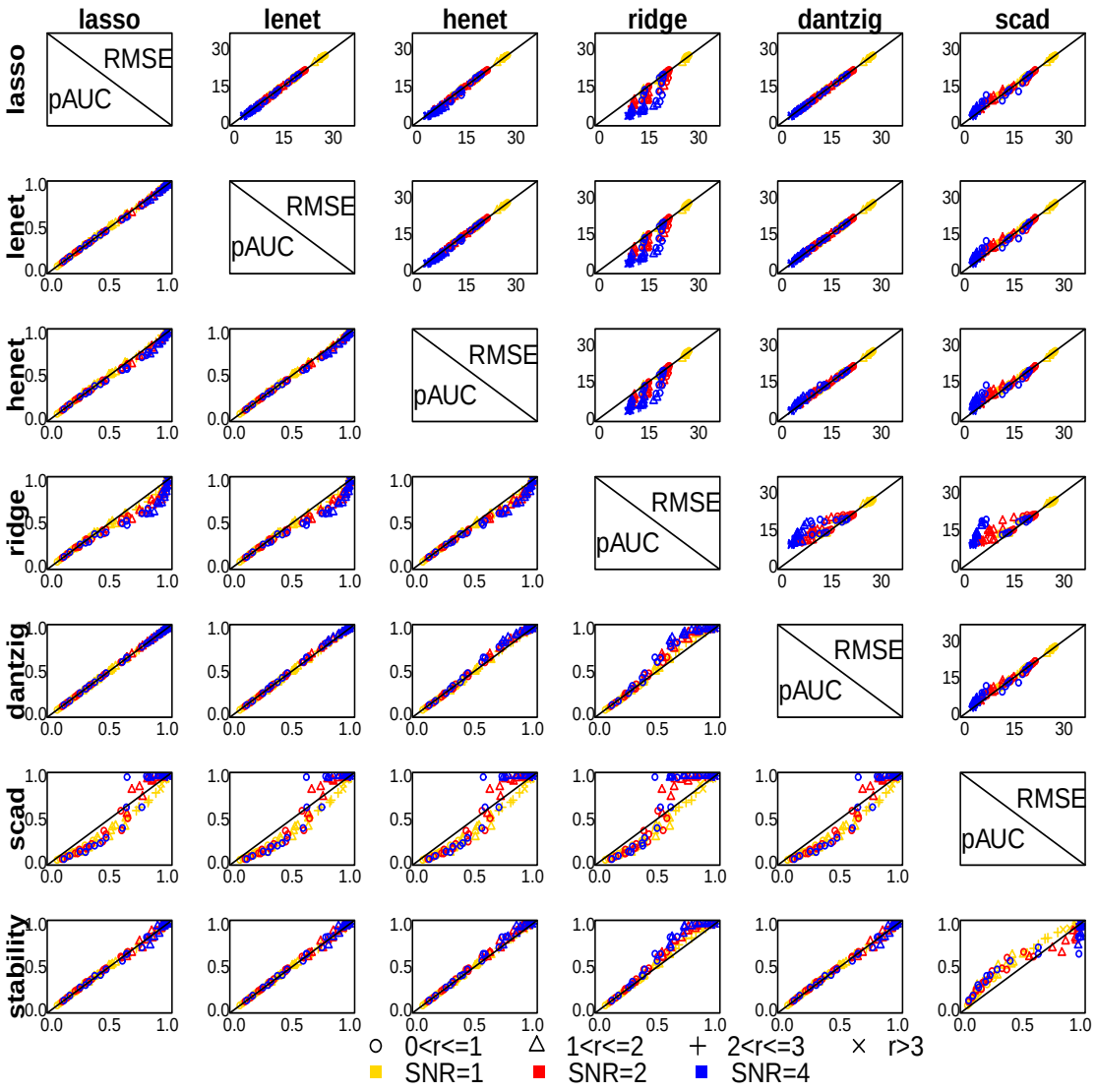


Figure 2.A4: A comparison of method performance in independence design scenarios: ranking and prediction. The upper and lower triangular parts of the plot show prediction (RMSE) and ranking (pAUC) performance respectively. Each panel plots the ranking or prediction performance of one method versus the ranking or prediction performance of another method. Row and column labels indicate which method is plotted on the y-axis and x-axis respectively. Each data point within a panel corresponds to an independence design scenario with colour indicating SNR and symbol representing the value of the rescaled sample size r (categorised). Note that prediction performance is not assessed for Stability Selection.

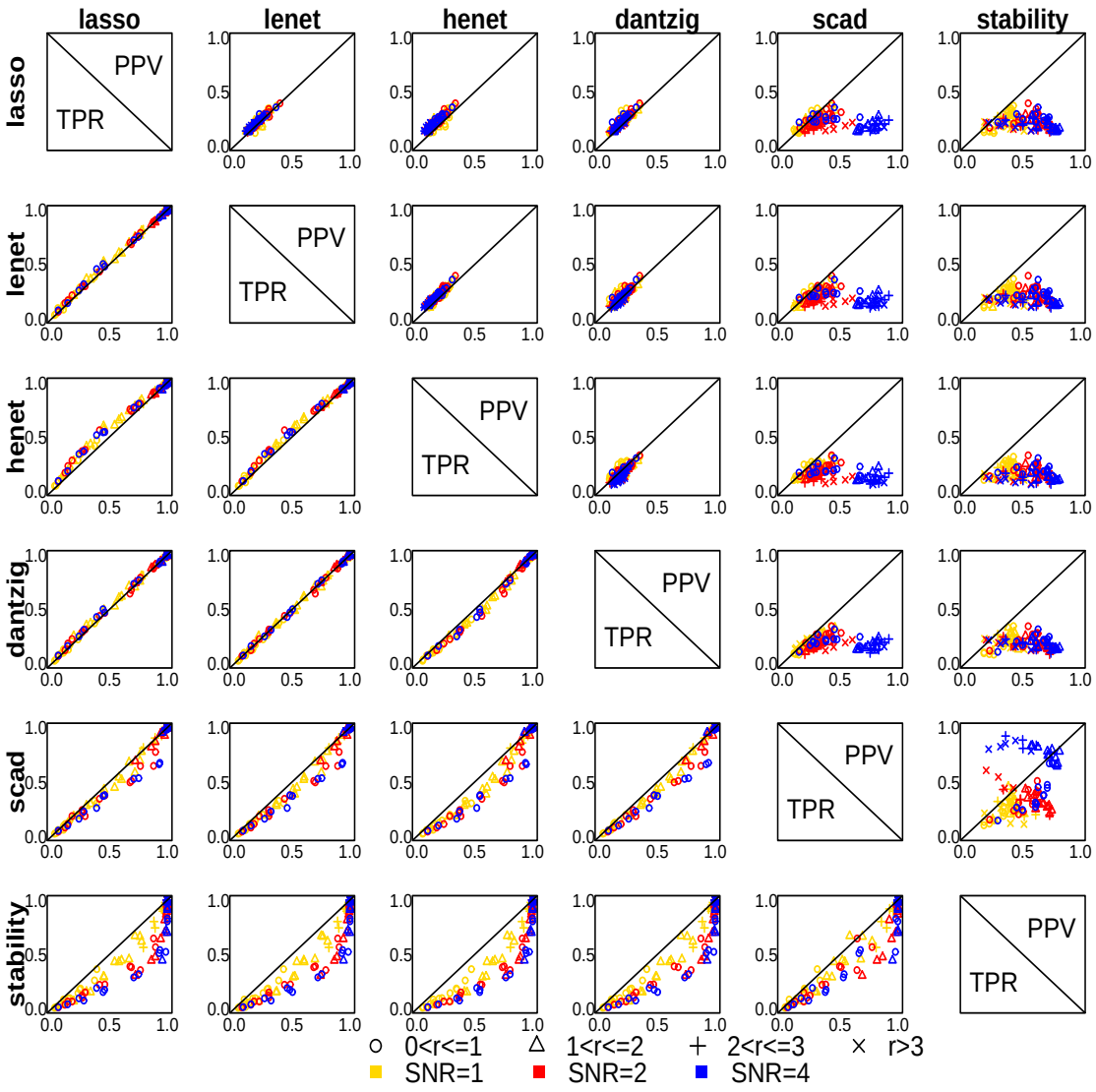


Figure 2.A5: A comparison of method performance in independence design scenarios: selection. The upper and lower triangular parts of the plot show PPV and TPR selection metrics respectively. Each panel plots PPV or TPR of one method versus PPV or TPR of another method. Row and column labels indicate which method is plotted on the y-axis and x-axis respectively. Each data point within a panel corresponds to an independence design scenario with colour indicating SNR and symbol representing the value of the rescaled sample size r (categorised).

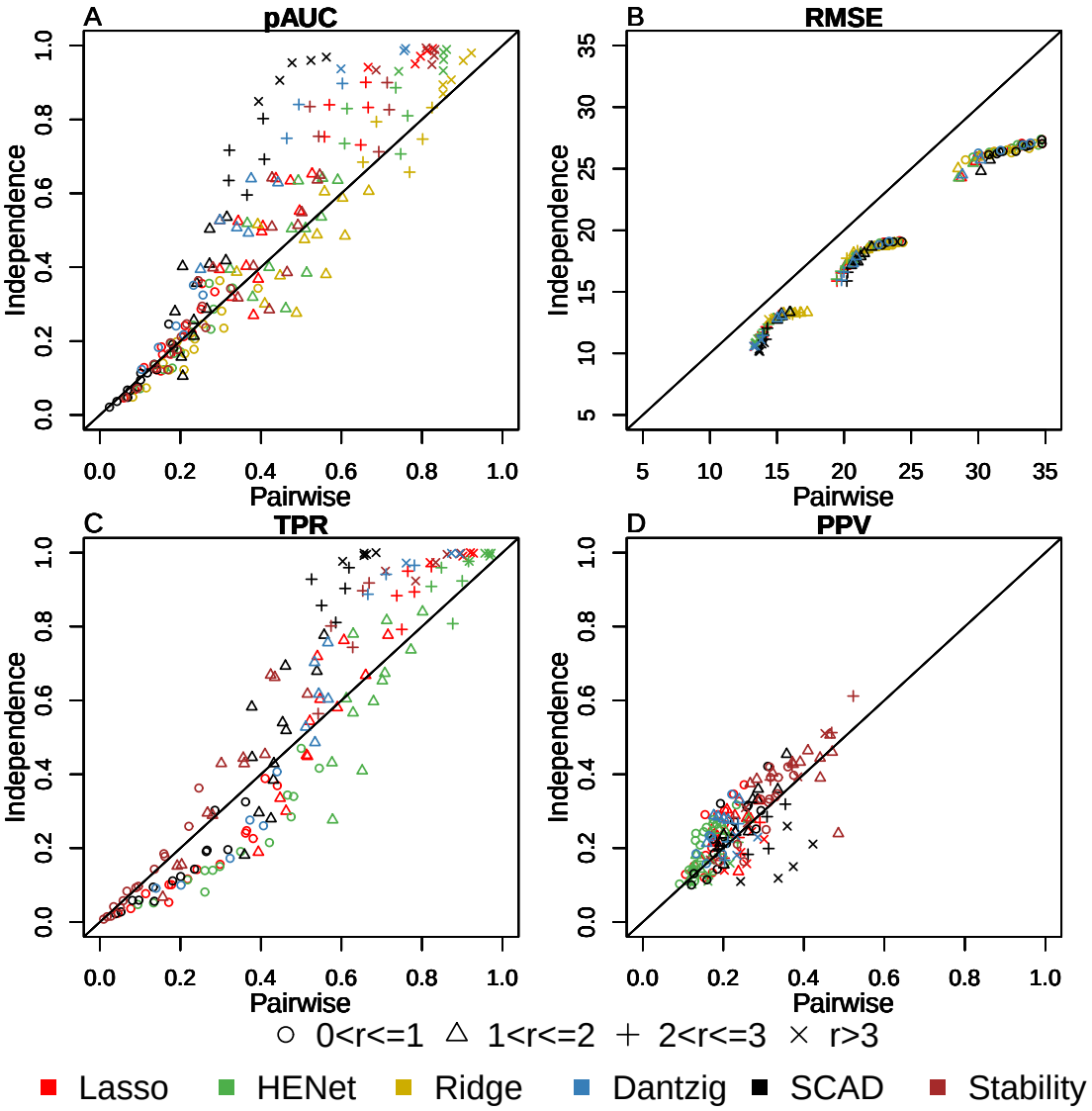


Figure 2.A6: Influence of correlation on ranking (A), prediction (B) and selection (C,D) performance, relative to the independence design. As Figure 2.5, but with SNR=1 (instead of SNR=2). Performance in the independence design is plotted against performance in the pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 10$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol).

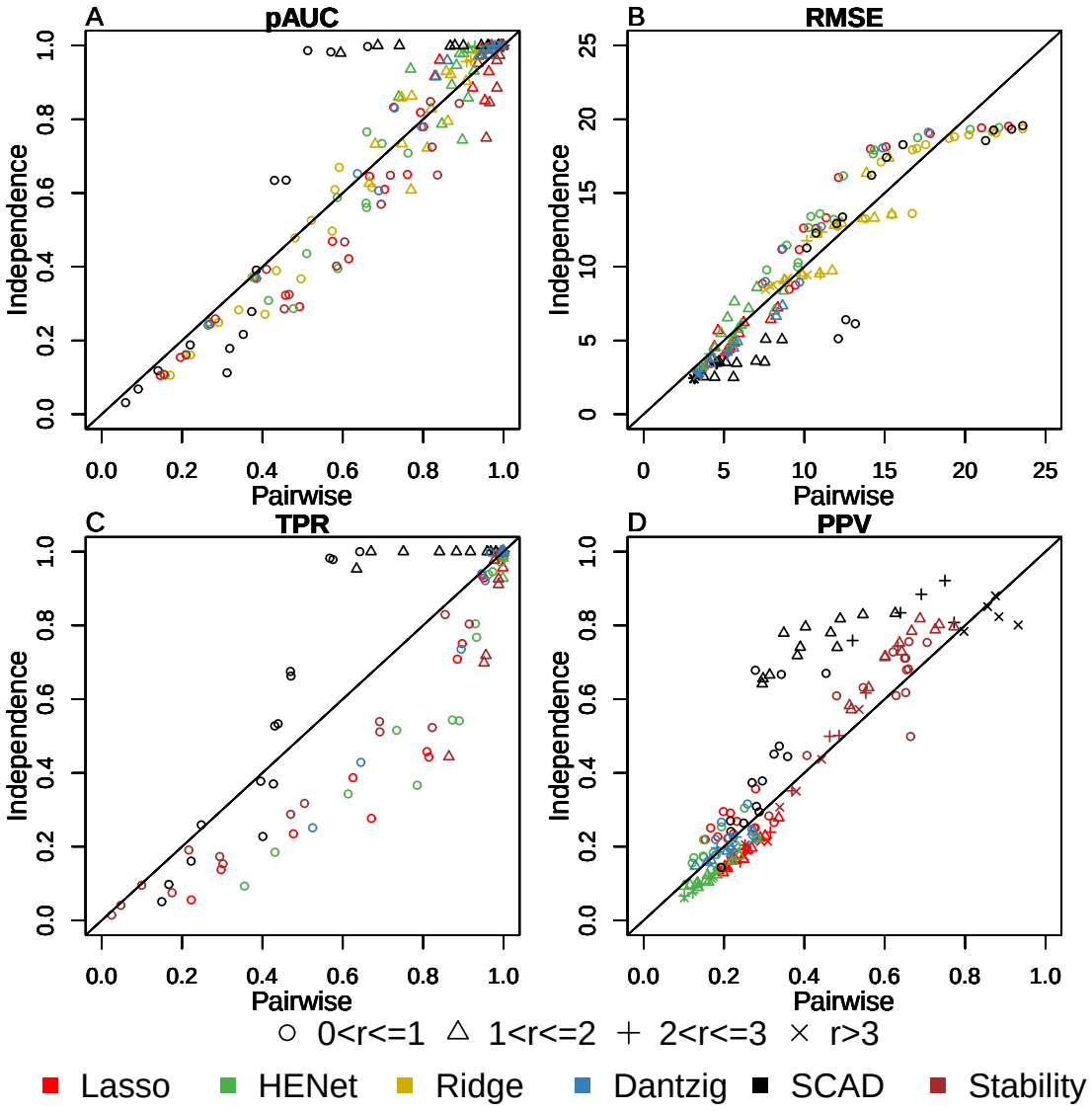


Figure 2.A7: Influence of correlation on ranking (A), prediction (B) and selection (C,D) performance, relative to the independence design. As Figure 2.5, but with SNR=4 (instead of SNR=2). Performance in the independence design is plotted against performance in the pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 10$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol).

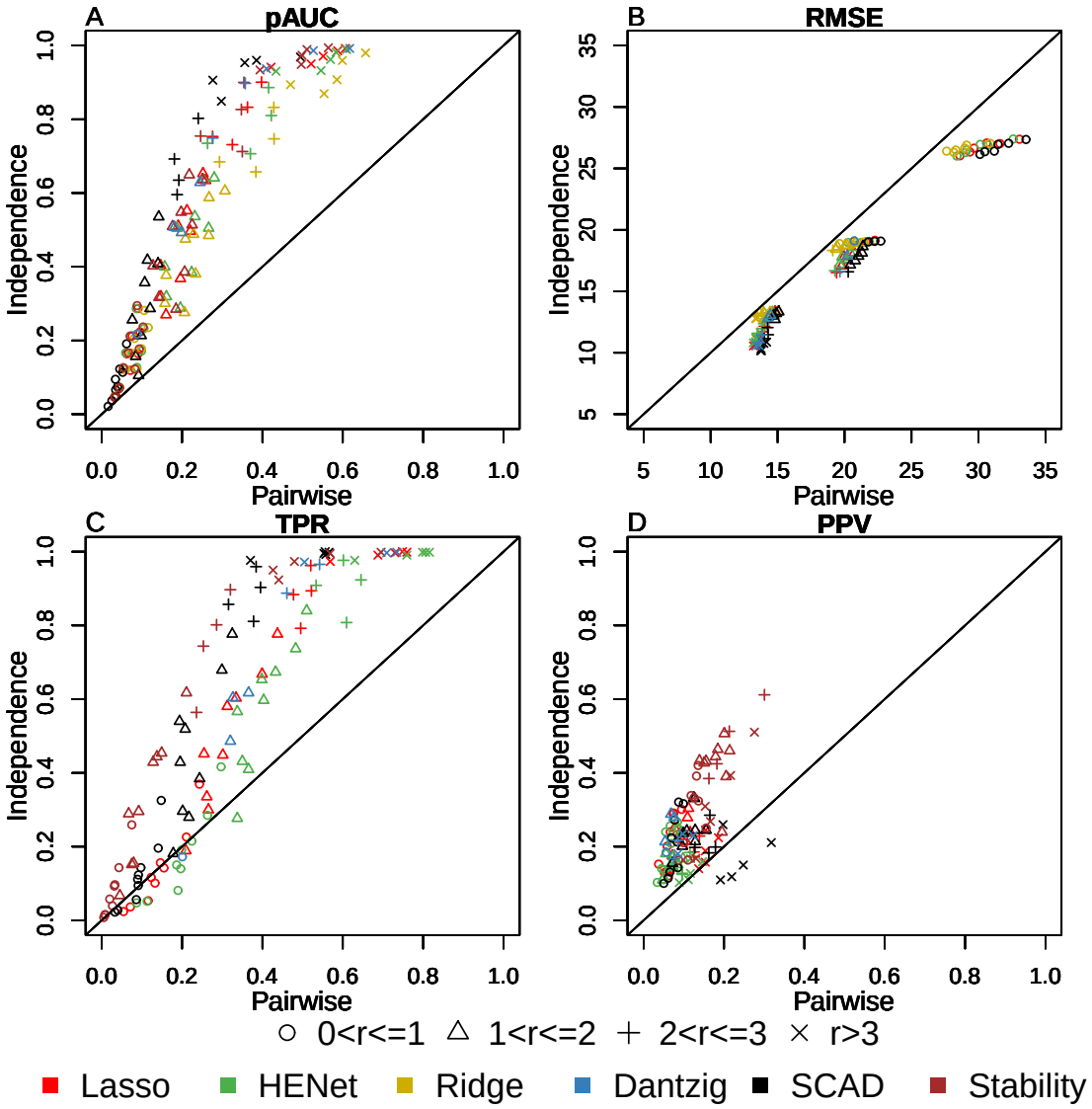


Figure 2.A8: Influence of correlation on ranking (A), prediction (B) and selection (C,D) performance, relative to the independence design. As Figure 2.6, but with SNR=1 (instead of SNR=2). Performance in the independence design is plotted against performance in the pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 100$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol).

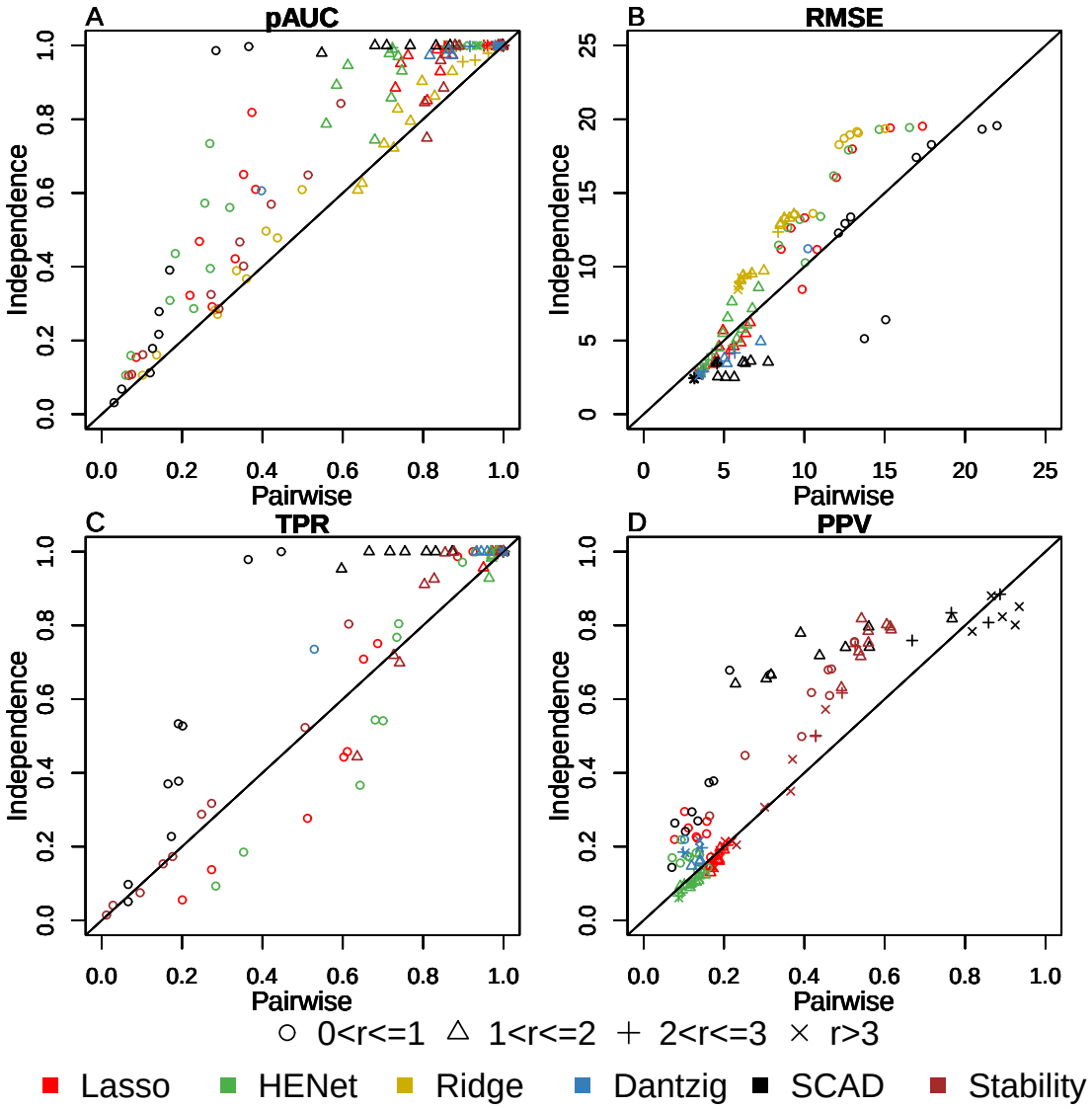


Figure 2.A9: Influence of correlation on ranking (A), prediction (B) and selection (C,D) performance, relative to the independence design. As Figure 2.6, but with SNR=4 (instead of SNR=2). Performance in the independence design is plotted against performance in the pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 100$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol).

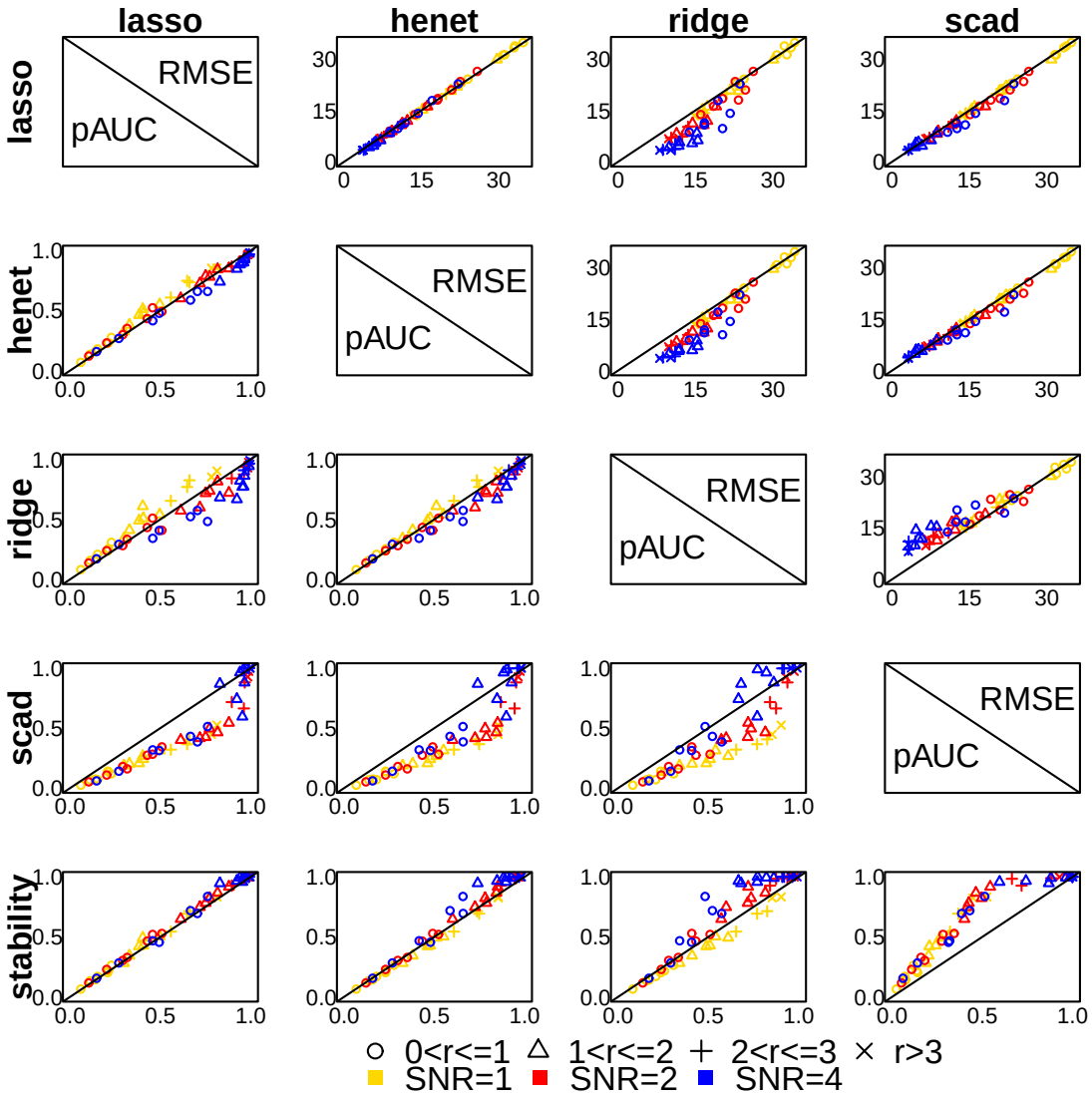


Figure 2.A10: A comparison of method performance for the pairwise correlation design scenarios with $\rho = 0.7$, $s_0^B = 2$ and $p^B = 10$: ranking and prediction. The upper and lower triangular parts of the plot show prediction (RMSE) and ranking (pAUC) performance respectively. Each panel plots the ranking or prediction performance of one method versus the ranking or prediction performance of another method. Row and column labels indicate which method is plotted on the y-axis and x-axis respectively. Each data point within a panel corresponds to a correlation design scenario (with $\rho = 0.7, s_0^B = 2, p^B = 10$) with colour indicating SNR and symbol representing the value of the rescaled sample size r (categorised). Note that prediction performance is not assessed for Stability Selection, and LENet and Dantzig are not shown.

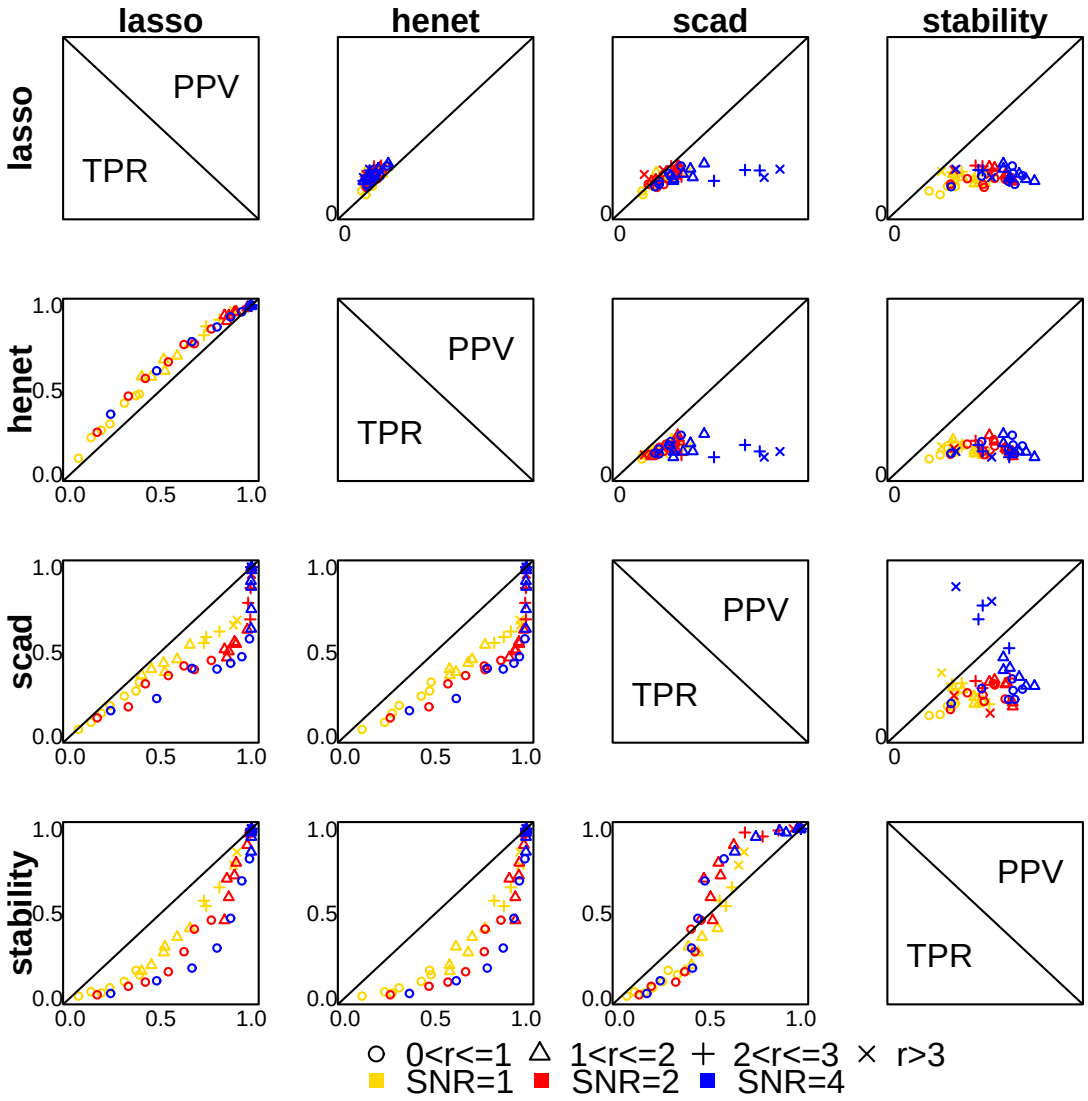


Figure 2.A11: A comparison of method performance for the pairwise correlation design scenarios with $\rho = 0.7$, $s_0^B = 2$ and $p^B = 10$: selection. The upper and lower triangular parts of the plot show PPV and TPR selection metrics respectively. Each panel plots PPV or TPR of one method versus PPV or TPR of another method. Row and column labels indicate which method is plotted on the y-axis and x-axis respectively. Each data point within a panel corresponds to a correlation design scenario (with $\rho = 0.7, s_0^B = 2, p^B = 10$) with colour indicating SNR and symbol representing the value of the rescaled sample size r (categorised). LENet and Dantzig are not shown.

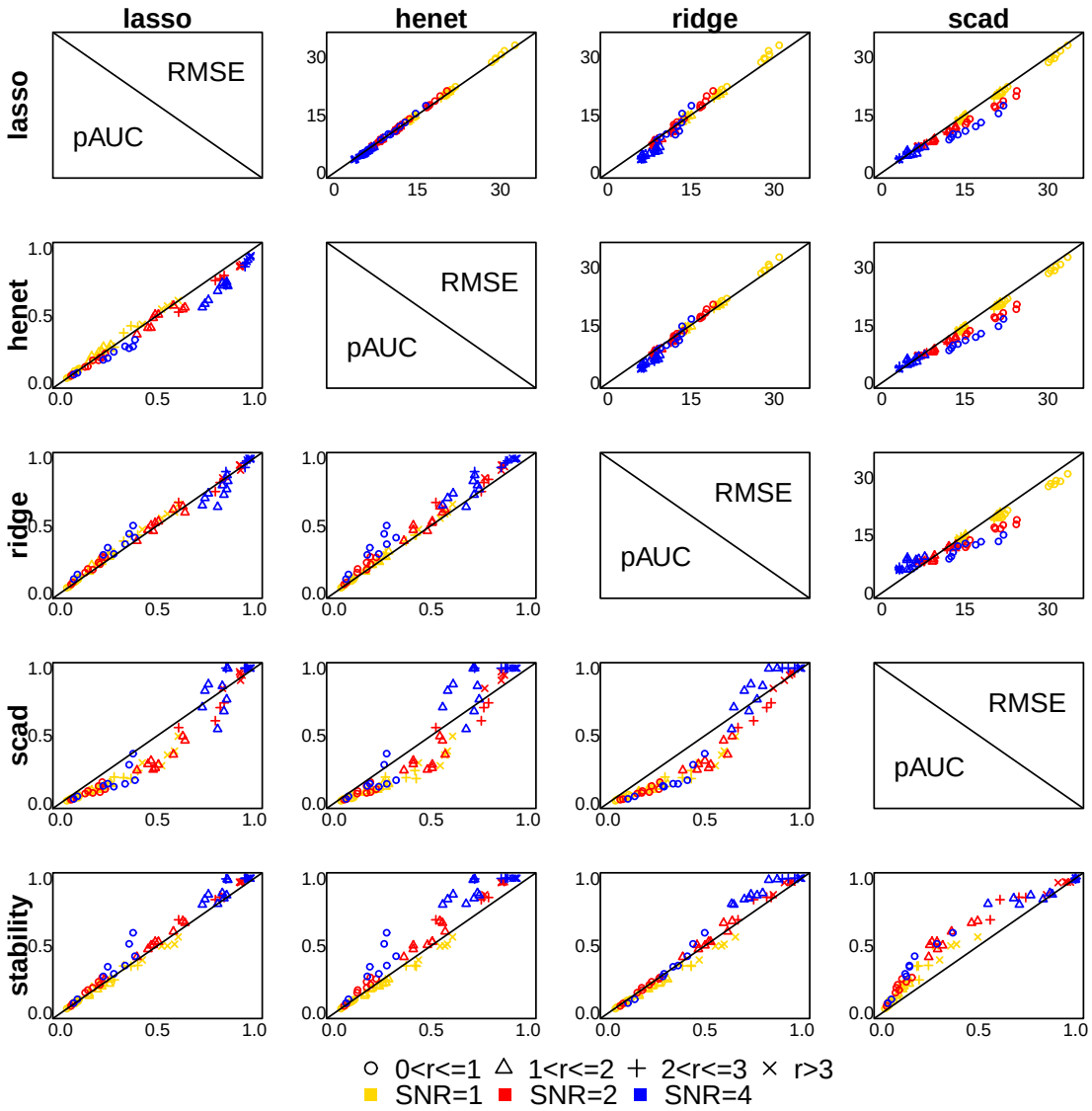


Figure 2.A12: A comparison of method performance for the pairwise correlation design scenarios with $\rho = 0.7$, $s_0^B = 2$ and $p^B = 100$: ranking and prediction. As Figure 2.A10, but with $p^B = 100$ (instead of $p^B = 10$).

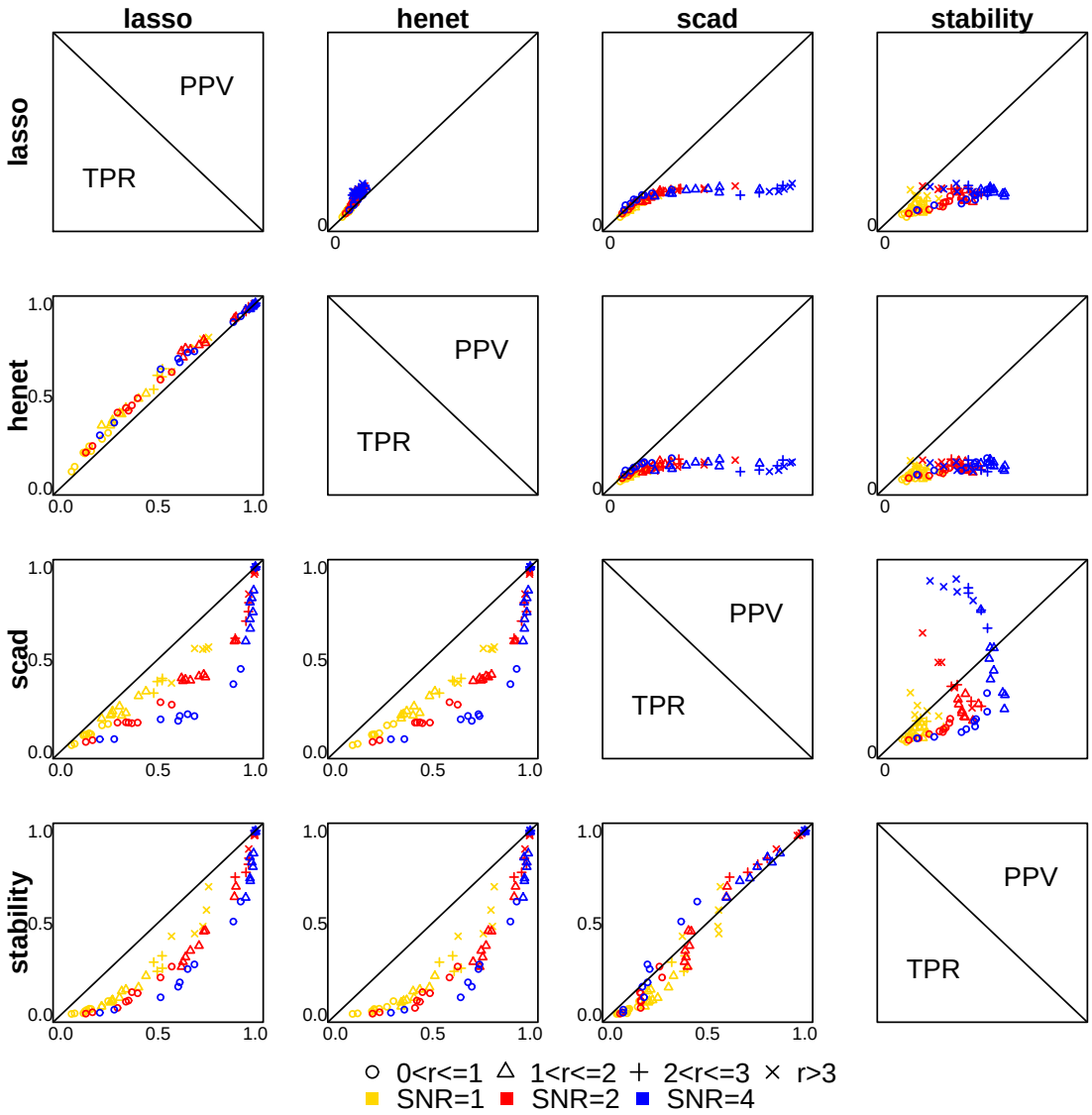


Figure 2.A13: A comparison of method performance for the pairwise correlation design scenarios with $\rho = 0.7$, $s_0^B = 2$ and $p^B = 100$: selection. As Figure 2.A11, but with $p^B = 100$ (instead of $p^B = 10$).

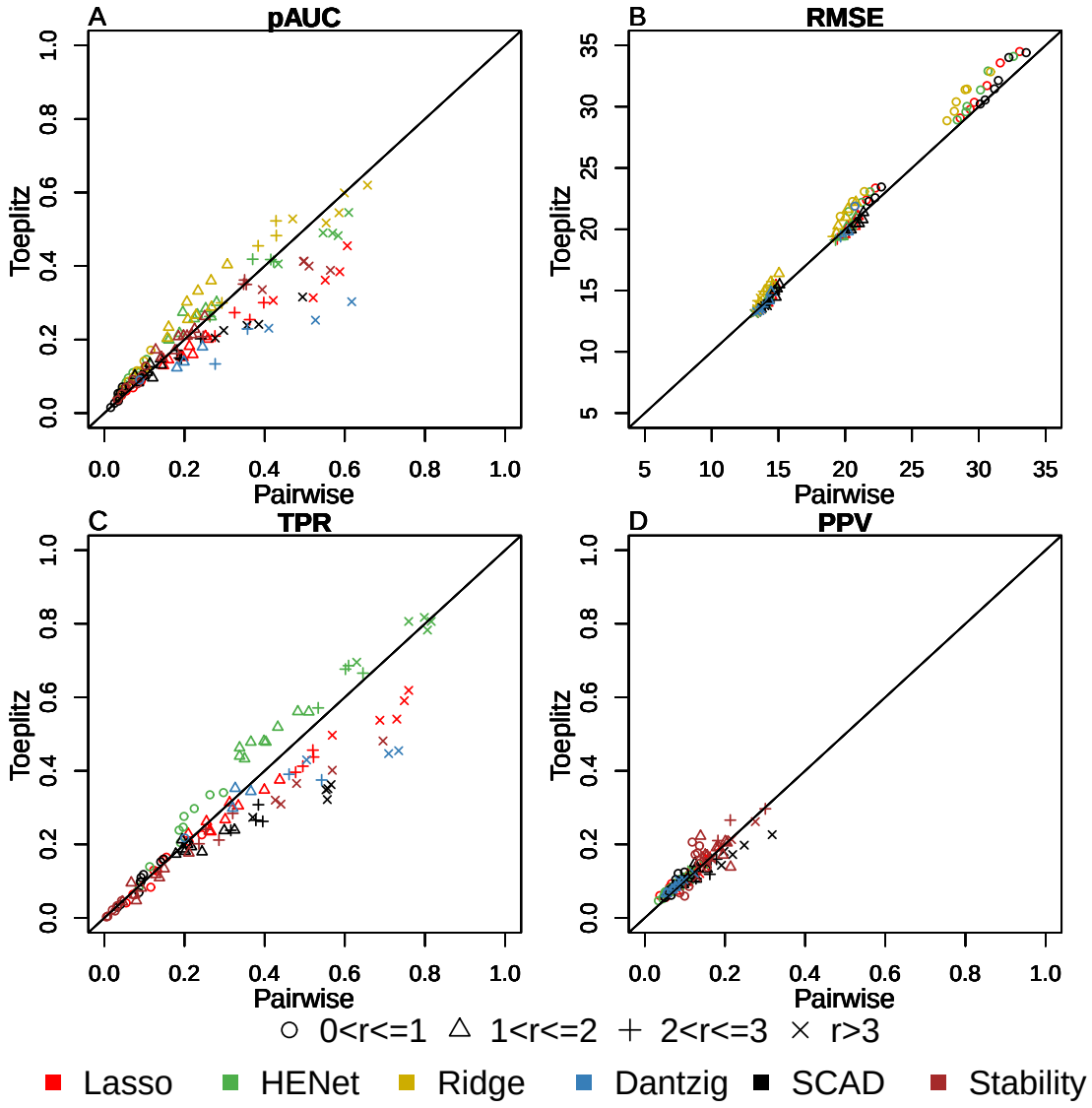


Figure 2.A14: Comparison between Toeplitz correlation and pairwise correlation designs for ranking (A), prediction (B) and selection (C,D) performance. As Figure 2.13, but with SNR=1 (instead of SNR=2). Performance in the Toeplitz correlation design (see Methods) is plotted against performance in the corresponding pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 100$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol).

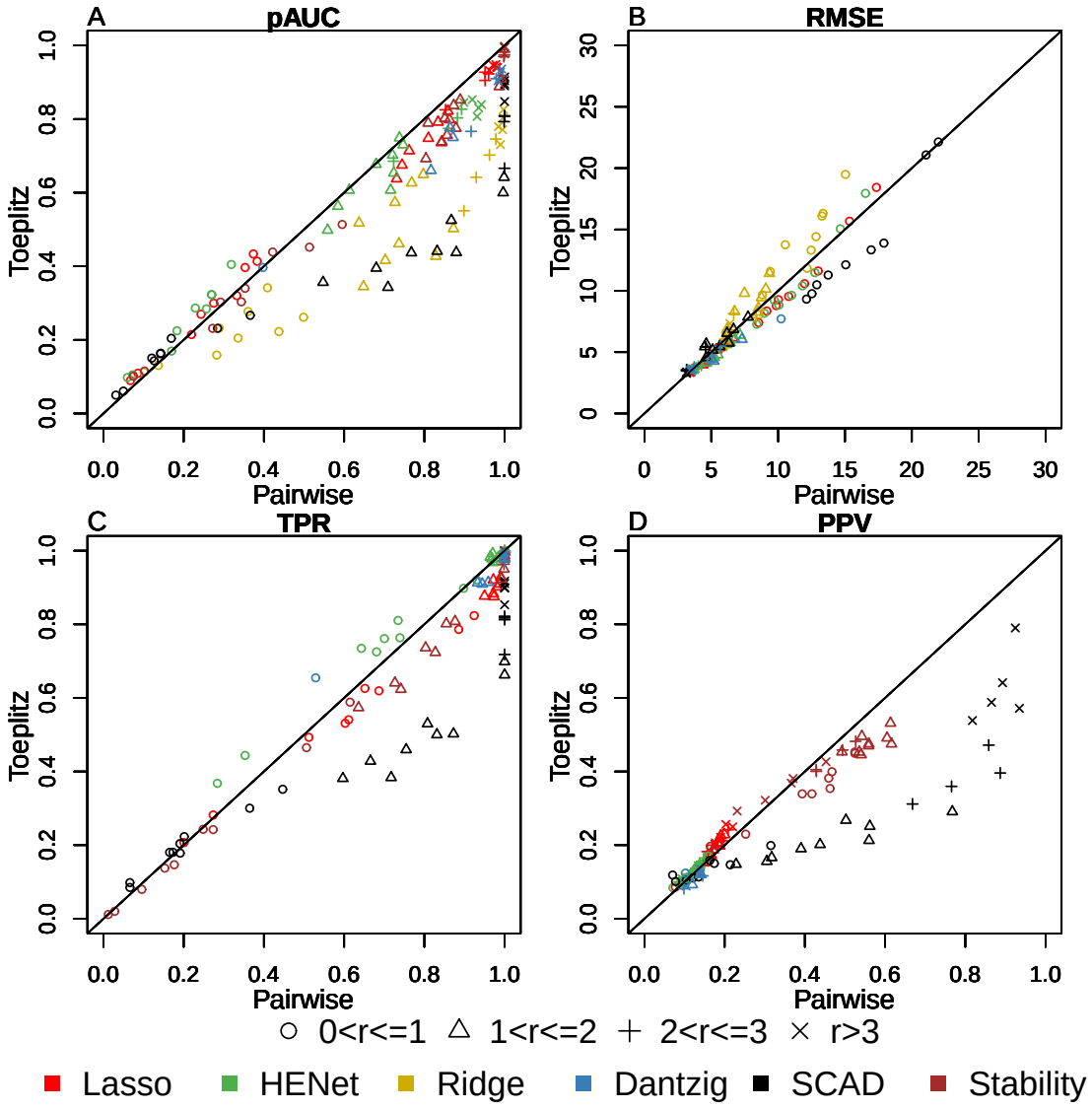


Figure 2.A15: Comparison between Toeplitz correlation and pairwise correlation designs for ranking (A), prediction (B) and selection (C,D) performance. As Figure 2.13, but with SNR=4 (instead of SNR=2). Performance in the Toeplitz correlation design (see Methods) is plotted against performance in the corresponding pairwise correlation design with $\rho = 0.7, s_0^B = 2$ and $p^B = 100$. Each point corresponds to a method (indicated by colour) and a single (n, p, s_0) triplet (the resulting value of the rescaled sample size r is indicated by symbol).

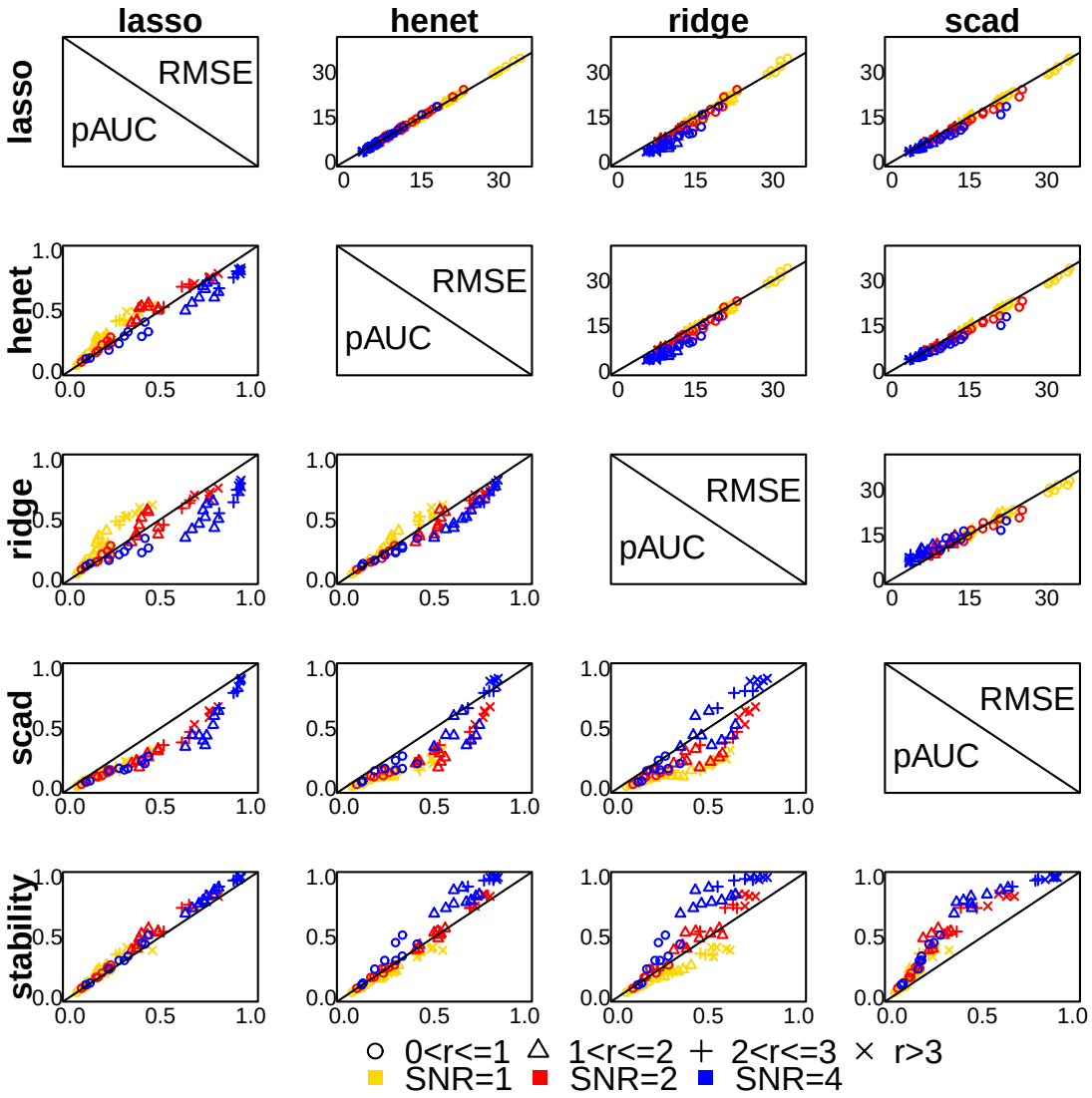


Figure 2.A16: A comparison of method performance for the Toeplitz correlation design: ranking and prediction. As Figure 2.A12, but with Toeplitz correlation instead of pairwise correlation.

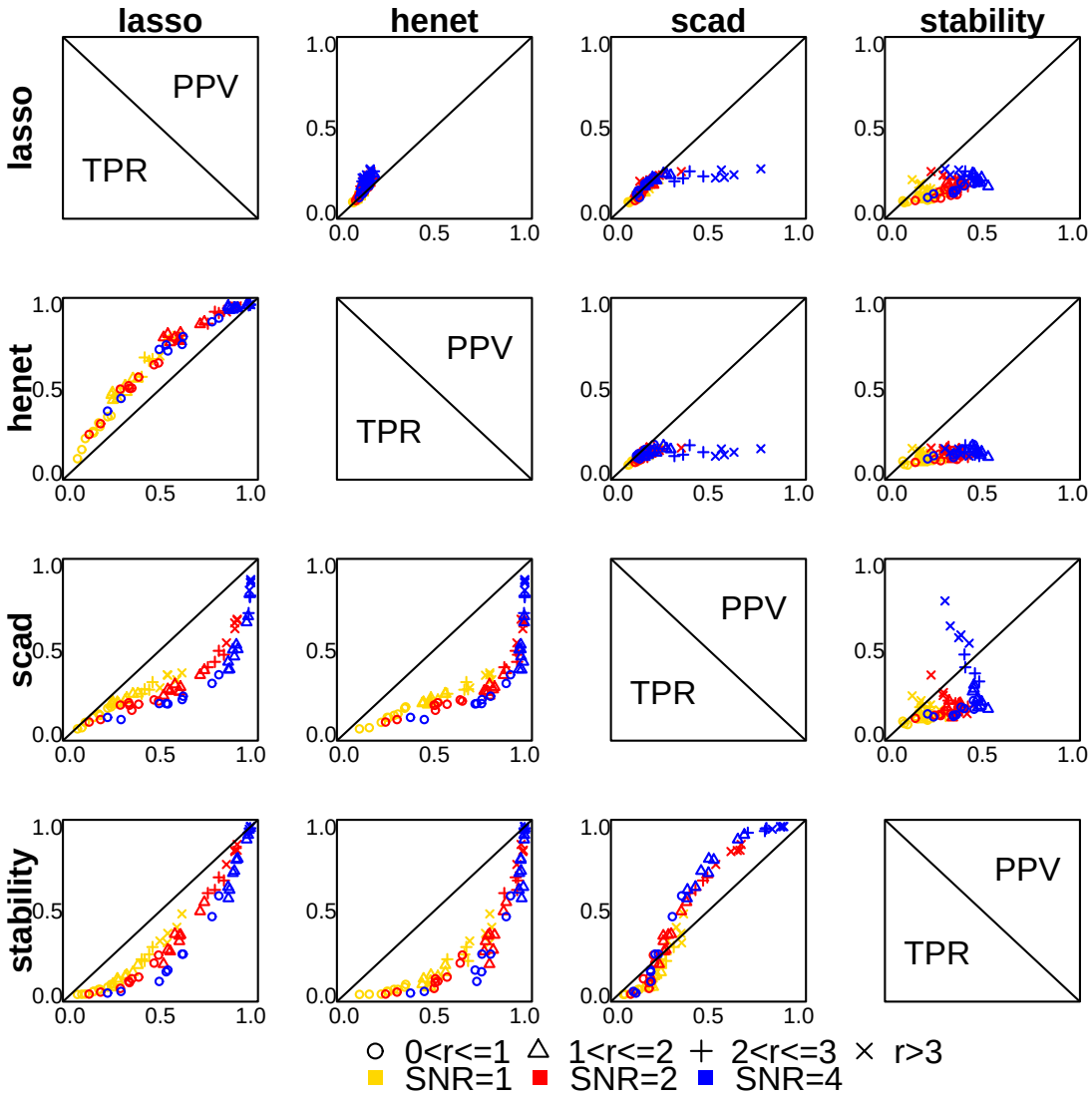


Figure 2.A17: A comparison of method performance for the Toeplitz correlation design: selection. As Figure 2.A13, but with Toeplitz correlation instead of pairwise correlation.

Chapter 3

Systematic comparison: additional investigations

In this chapter we extend the main simulations from Chapter 2 to explore additional model assumptions and parameter specifications. Section 3.1 explores sensitivity of Stability Selection to its tuning parameters. Section 3.2 investigates the ability of methods to detect weak signals when coefficients are heterogeneous. Section 3.3 makes detailed comparison between Lasso, Elastic Net and Ridge Regression in correlation design. Section 3.4 compares prediction performance when underlying model is not sparse, and Section 3.5 aims to find out the influence of non-Gaussian error on relative performance. Datasets are generated using linear model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, and different designs are simulated as described in Section 2.2.1. All results in this section are averaged across 100 replicates.

3.1 Stability Selection tuning parameters

Stability Selection has several tuning parameters: the subsample size $\tilde{n}$, an upper bound $\tilde{V}$ for $\mathbb{E}[V]$ (the expected number of false positives), and either a threshold π_{thr} on the selection probabilities or a set of regularization parameters to consider Λ (see Section 1.6.1). Making appropriate choices for these parameters is non-trivial. Here, we explore the effects of varying $\tilde{n}$, $\tilde{V}$ and π_{thr} on selection performance.

We simulated data (as described in Section 2.2.1) with SNR=2, n=200, p=1000

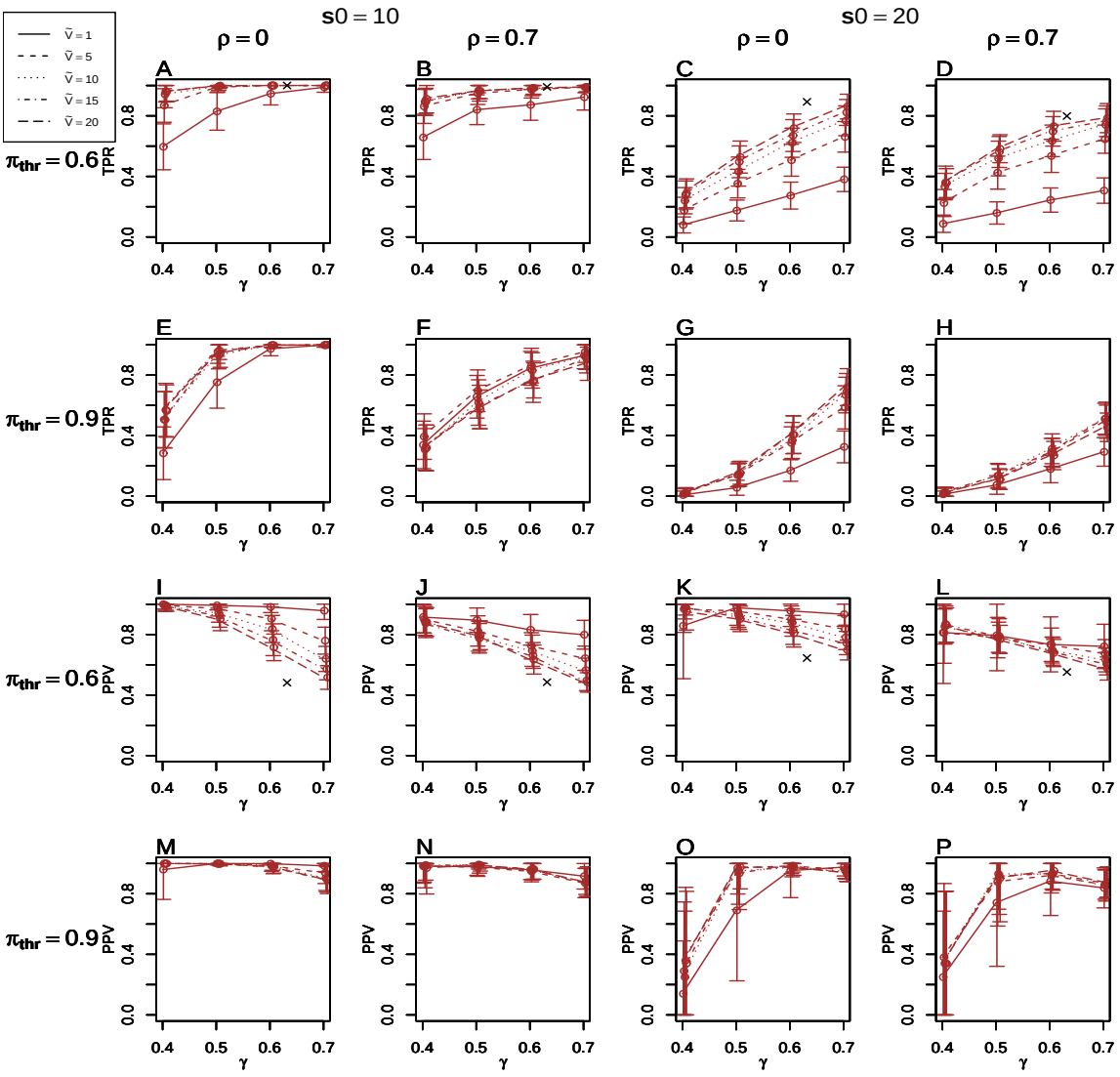


Figure 3.1: Stability Selection tuning parameter sensitivity. TPR (top two rows) and PPV (bottom two rows) versus the subsample proportion γ for two values of the threshold probability π_{thr} (rows) and five values of $\tilde{V}$, the upper bound for the expected number of false positives (indicated by line type within each panel). Each column corresponds to a different simulation scenario: the independence design with $n = 200, p = 1000$, SNR=2 and $s_0 = 10$ (first column) or $s_0 = 20$ (third column) or the corresponding pairwise correlation design scenarios with p^B=10, $s_0^B = 2$ and $\rho = 0.7$ (second and fourth columns). The panels corresponding to $\pi_{thr} = 0.6$ each contain a black cross that shows the performance observed in the main simulations where $\gamma = 0.632$, $\pi_{thr} = 0.6$ and there was no explicit false positive control $\tilde{V}$.

and s_0=10 or 20 (giving $r = 2.90$ or 1.45 respectively) for the independence design, and the pairwise correlation design with p^B=10, $s_0^B = 2$ and $\rho = 0.7$. We applied Stability Selection with all possible combinations of the following tuning parameter values: $\tilde{V} \in \{1, 5, 10, 15, 20\}$, $\pi_{thr} \in \{0.6, 0.9\}$ and $\tilde{n} = \lfloor n\gamma \rfloor$ where $\gamma \in \{0.4, 0.5, 0.6, 0.7\}$ is the subsample proportion.

Results are shown in Figure 3.1. In general, as $\tilde{V}$ or γ increase, or π_{thr} decreases, the number of selected variables increases, resulting in higher TPR, but lower PPV. An exception is for $s_0 = 20$, where, for the most conservative choices of the parameters ($\gamma = 0.4, \tilde{V} = 1$ and $\pi_{thr} = 0.9$), in addition to a very poor TPR, PPV is low on average and very unstable across iterations (see Fig. 3.1G, H, O and P). Here, selection is too stringent and the majority of signals are missed. An increase in $\tilde{V}$ and γ, and decrease in π_{thr} leads to substantial improvements in both TPR and PPV. When the underlying model size is smaller ($s_0 = 10$), the most conservative parameter choices are again suboptimal in terms of performance, but the same is also true for the least conservative choices ($\gamma = 0.7, \tilde{V} = 20$ and $\pi_{thr} = 0.6$; Fig. 3.1I and J). However, in the scenarios considered here, being too stringent seems to have a more deleterious effect on performance than being too lenient.

Results from the main simulations, where we set $\tilde{n} = \lfloor 0.632n \rfloor$, $\pi_{thr} = 0.6$ and had no explicit false positive control $\tilde{V}$ (i.e. the full range of regularization parameters Λ was considered; see Section 2.2.2), are indicated by crosses in panels A-D and I-L of Figure 3.1. Performance in the main simulations is typically similar to that of the largest $\tilde{V}$ considered here ($\tilde{V} = 20$), but with better TPR and worse PPV (except for $s_0 = 10$ where TPR is already optimal and so there is only a decrease in PPV).

3.2 Homogeneous coefficients

In Chapter 2, all non-zero coefficients were assigned the same value ($\beta = 3$). Here, we consider detection of signals with heterogeneous coefficients for three methods: Lasso, HENet and SCAD. We simulated data (for the independence design) as described in Section 2.2.1, except instead of s_0 active variables all having coefficient 3, half of them had coefficient β' and the other half had coefficient

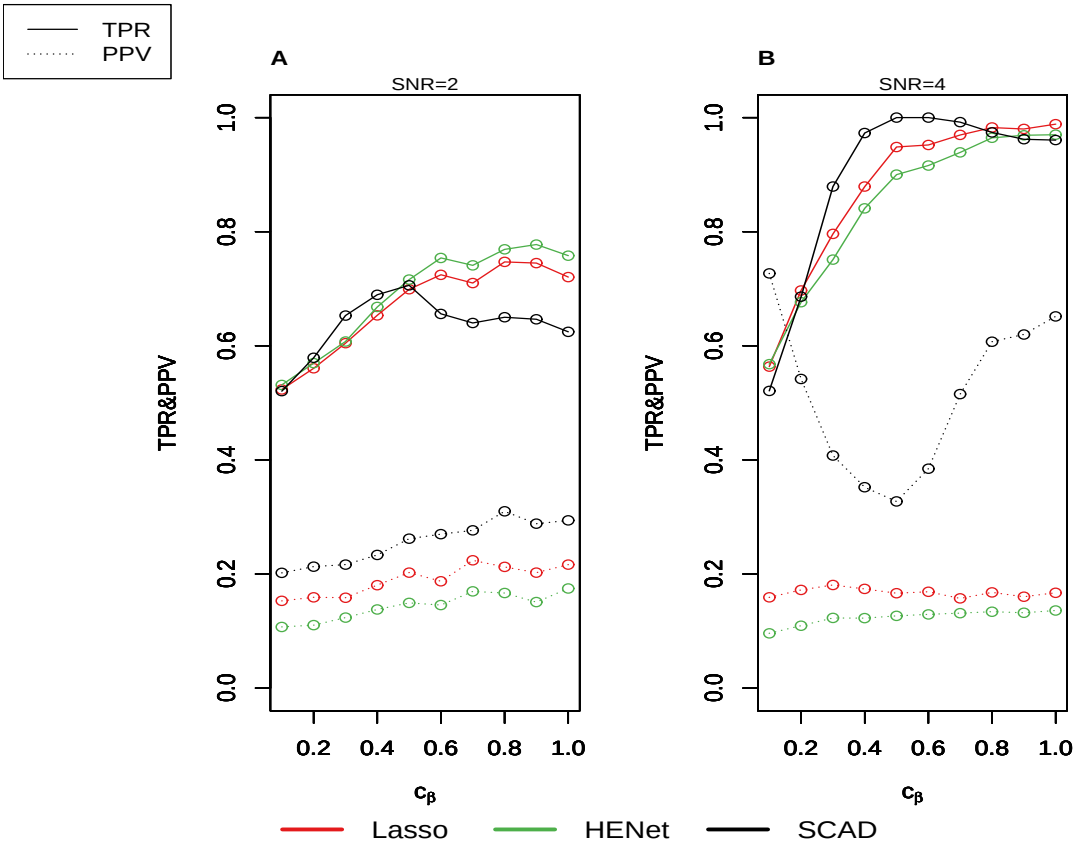


Figure 3.2: Influence of heterogeneous regression coefficients on selection performance. TPR (solid lines) and PPV (dotted lines) are plotted against the coefficient scaling factor c_β for the independence design with $(n, p, s_0) = (300, 4000, 40)$ and SNR=2 (A) or SNR=4 (B). In the data-generating linear model, half of the signals have coefficient β' and the other half have coefficient $c_\beta\beta'$ (see text for details). Note that $c_\beta = 1$ gives the main simulation set-up with homogeneous coefficients. Line colour indicates method.

$c_\beta\beta'$ where $c_\beta \in [0,1]$. We chose $\beta' = \sqrt{18/(1+{c_\beta}^2)}$ such that with fixed SNR, $\mathbb{E}(\sigma^2)$ remains the same as in the homogeneous β's case. Note that $c_\beta = 1$ gives the main simulation set-up with homogeneous coefficents. Informed by the main simulations, we set $n = 300$, $s_0 = 40$, $p = 4000$ and SNR=2 or 4, guaranteeing that when non-zero coefficients all take the same value, we are in a relatively "easy" scenario where the majority of the signals can be detected.

Figure 3.2 shows the effect of heterogeneous coefficients on selection for $c_\beta \in \{0.1, 0.2, \ldots, 1\}$. As c_β decreases, signals with smaller coefficients are less likely to be detected, resulting in a decrease in TPR. All methods fail to detect the very weak signals when c_β=0.1 (i.e. only the stronger 50% of the signals are detected giving TPR≈0.5). Consistent with the main simulations, SCAD has better false positive control (higher PPV) than Lasso and Elastic Net when SNR is large, and this is especially the case when c_β is near 0.1 or 1 (contrast black dotted line with red and green dotted lines in Fig. 3.2B). The "U" shape of the SCAD PPV curve here is due to the fact that bias is largest when c_β is moderate, which leads to selection of more variables to compensate (SCAD is known to be nearly unbiased for strong signals; for large c_β all signals are relatively strong, while for small c_β the $s_0/2$ weaker signals have such a small influence that the underlying model is well-approximated by a model with $s_0/2$ strong signals and no weak signals). In contrast, Lasso and Elastic Net are biased estimators, so their PPV are not as affected. SCAD also seems to have higher power to detect the weaker signals when SNR is large and c_β is moderate (see solid lines in Fig. 3.2B). However, as observed in the main simulations, SCAD is more sensitive to SNR and so is less competitive in "harder" scenarios (SNR=2; Figure 3.2A). Relative performance of Lasso and Elastic Net is consistent with the homogeneous coefficient case ($c_\beta = 1$). Note that in the independence design scenario with SNR=4 considered here, Lasso has slightly higher TPR than HENet, but in the majority of independence design scenarios the opposite relationship is observed (see Fig. 2.A5).

3.3 l_2 penalty in correlation design

In Chapter 2 we saw that an l_2 penalty does not offer advantages unless in extreme scenarios where strong multicollinearity exists, i.e., many relevant variables

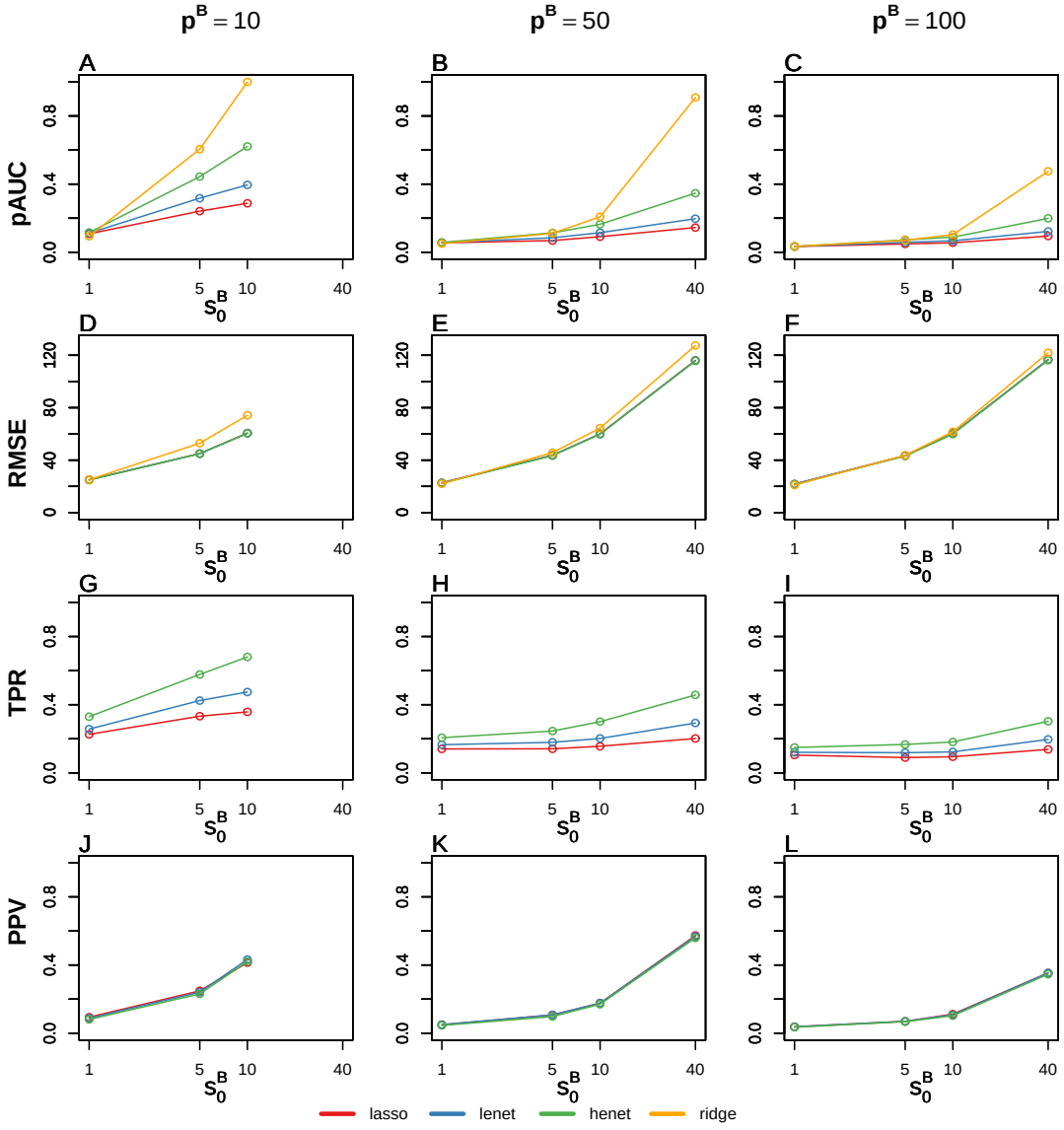
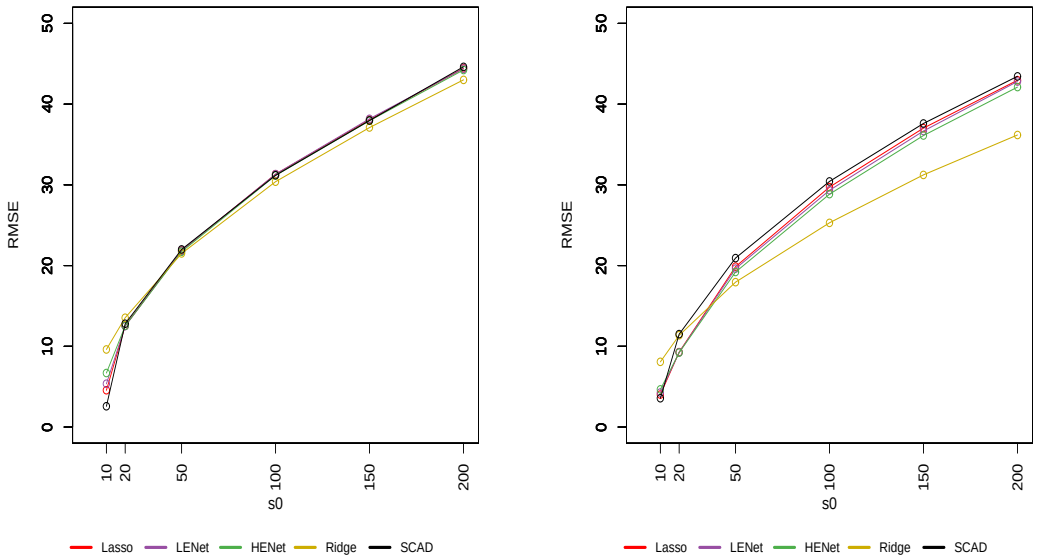


Figure 3.3: Influence of p^B and s_0^B in strong pairwise correlation design. Performance metrics are plotted against s_0^B with $(n, p, s_0) = (300, 4000, 40)$ and SNR=1, $\rho = 0.9$. Note that points corresponding to scenarios where $p/p^B * s_0^B < s_0$ are excluded. The x-axis is plotted on a log scale to facilitate presentation. Line colour indicates method.

are strongly correlated with each other. The benefit of an l_2 penalty was mainly observed for true positive rate and ranking. In this section we perform a more detailed simulation study on how block size p^B and number of relevant variables per block s_0^B influence the relative performance of Lasso, Elastic Net and Ridge Regression. We simulate data for pairwise correlation design as described in Section 2.2.1, and fix $n = 300, p = 4000, s_0 = 40$, SNR=1, and $\rho = 0.9$. We vary p^B to be 10, 50 or 100, and vary s_0^B to be 1, 5, 10 or 40. Figure 3.3 shows ranking, prediction and selection performance of Lasso, LENet, HENet and Ridge Regression. The relative performance of Lasso, Elastic Net and Ridge Regression depends on the relative magnitude of p^B and s_0^B. For pAUC and TPR, l_2 penalty seems to bring more significant benefit when s_0^B/p^B is larger. Fixing p^B, larger s_0^B boosts ranking of Ridge Regression more than Elastic Net, and has the smallest positive impact on Lasso (Fig. 3.3 A, B, C). Similar argument is true for TPR between Elastic Net and Lasso (Fig. 3.3 G, H, I). On the other hand, when s_0^B is fixed, the larger p^B is, the smaller benefit of l_2 penalty on pAUC and TPR is observed (compare performance metrics of fixed s_0^B across Fig. 3.3 A, B, C and Fig. 3.3 G, H, I). RMSE and PPV are similar among methods, and consistent with previous conclusions, Ridge Regression is typically less competitive for prediction, especially when s_0^B/p^B is large (Fig. 3.3 D, E, F).

3.4 Ridge-favourable settings

In Chapter 2 we saw that Ridge Regression is in general less competitive than sparse penalised regression methods when only a small proportion of variables are relevant. In this section we investigate the relative prediction performance in more pro-ridge scenarios, where s_0 is large, possibly greater than n. This type of data is common in biological studies. For example, a typical microarray study may have 10s of samples but potentially more genes associated with the response. We simulate data for independence and pairwise correlation designs as described in Section 2.2.1, and fix $n = 100, p = 2000$ and SNR=4. For correlation design, we fix block size $p^B = 10$, number of signals per block $s_0^B = 1$ and intra-block pairwise correlation $\rho = 0.9$. We vary s_0 to be 10, 20, 50, 100, 150 and 200, and evaluate prediction performance of Lasso, LENet, HENet, Ridge Regression and



(a) Independence design

(b) Pairwise correlation design

Figure 3.4: Comparison of prediction performance with different levels of sparsity. RMSE is plotted against s_0. $n = 100, p = 2000$, SNR=4. Left panel show results for independence design, and right panel show results for pairwise correlation design with $\rho = 0.9, p^B = 10$ and $s_0^B = 1$. Line colour indicates method.

SCAD in these scenarios. Figure 3.4 shows RMSE of methods with various s_0. For both designs, when s_0 is small ($s_0 = 10, 20$), Ridge Regression has the worst prediction performance. However, as s_0 increases, RMSE of all methods increases, but Ridge Regression is the least negatively affected. When s_0 is large enough, it achieves the best prediction performance among all methods, especially when $s_0 > n$, where methods such as Lasso will always be overly sparse. The advantage of Ridge Regression for prediction with large s_0 is more obvious in the pairwise correlation design. From results of systematic comparison study in Chapter 2 we see that Ridge Regression is also more robust in highly noisy or strongly correlated data. In practice, especially in certain fields of genetic studies, large s_0, high noise level and multicollinearity are common, where Ridge Regression should be given more credit.

3.5 Non-Gaussian error

In Chapter 2 we simulated data according to Model (1.1) where all model assumptions are satisfied. In practice some of the model assumptions can be violated. Non-Gaussian error is common in practice. Non-Gaussian errors occur if data contains outlying observations, or if heterogeneity exists when dataset is aggregated from (unknown) subpopulations. Since many physical processes are a summation of smaller processes which share the same distribution for random error, according to central limit theorem, the overall errors will follow Gaussian distribution; However, there are also processes where the errors are not sum of many smaller, identically distributed contributions, and in those cases, the errors can be non-Gaussian. We extend our comparison of methods to non-Gaussian error models to evaluate their robustness against this model assumption violation. We revisited a low correlation scenario from the TCGA ovarian cancer data analysis (Section 2.4) from Chapter 2, but with a non-Gaussian error distribution. We choose low correlation scenario with $n = 100$, $p = 1000$ and $s_0 = 10$. Instead of Gaussian error where $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I})$, 95% of error terms drawn from $N(0, \sigma^2)$ and the other 5% drawn from $N(0, (\tau\sigma)^2)$, with σ set such that SNR=4, and $\tau \in \{1, \ldots, 10\}$. In other words, the error terms are drawn from a mixture normal distribution, and the larger τ is, the further error terms deviate from normal distribution, where heavy-tailed errors are more influential ($\tau = 1$ is the standard Gaussian error scenario). Figure 3.5 shows method performance for all metrics. Performance of all methods deteriorates as non-normality increases. SCAD is the most affected and mirrors its previous behavior, with a transition in performance from best to worst as non-normality increases for ranking and prediction. Relative performance of other methods remains consistent as non-normality increases.

3.6 Discussion

In this chapter we perform further investigations based on the main simulations, to address some questions not covered in Chapter 2. Specifically, we consider the sensitivity of Stability Selection's selection performance against its tuning parameters, and the relative performance of methods when the set-up of linear model is

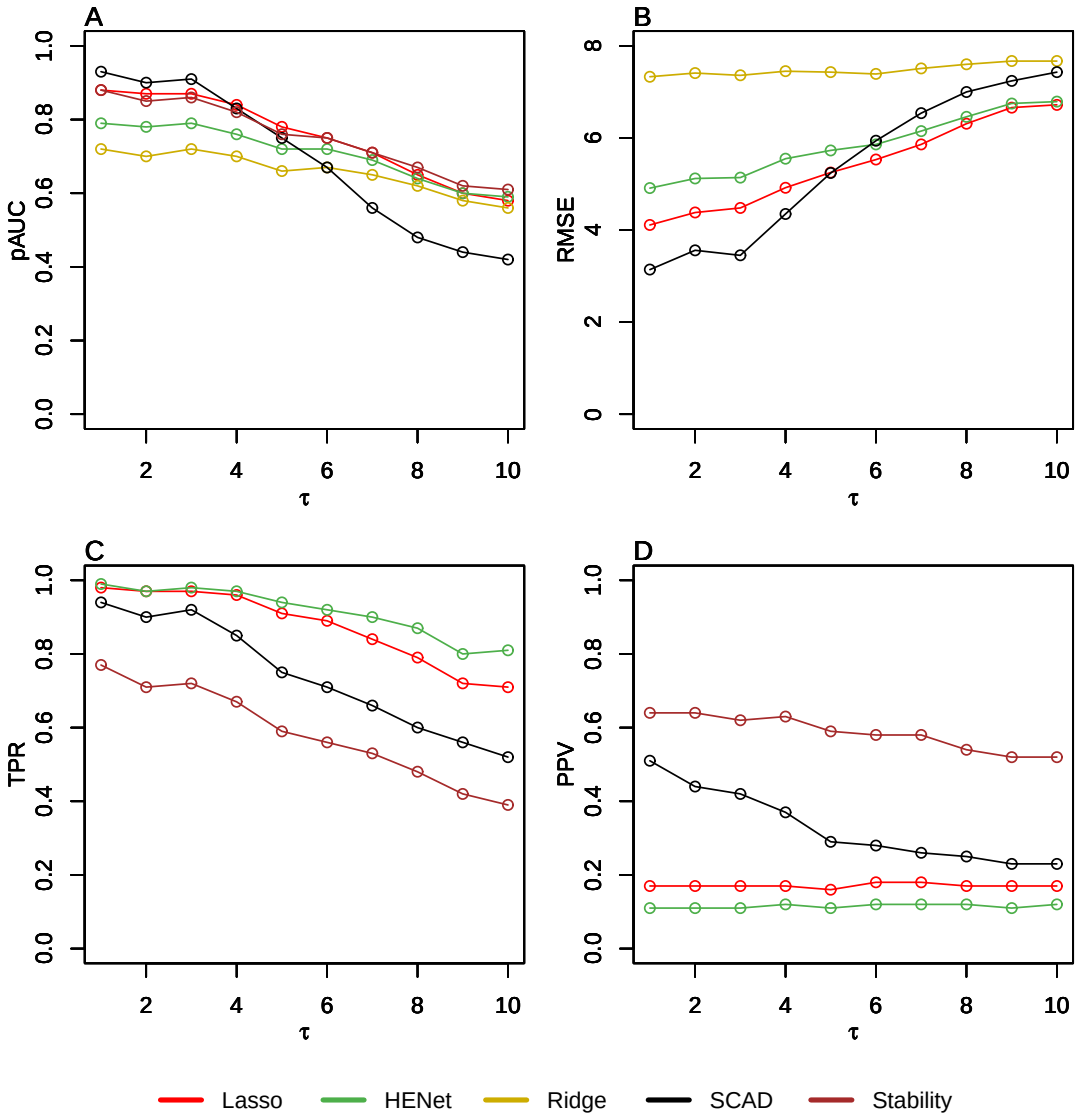


Figure 3.5: TCGA ovarian cancer expression data analysis: low correlation scenario with non-Gaussian error distribution. Ranking (A), prediction (B) and selection (C,D) performance is plotted against τ. Line colour indicates method.

different from the main simulation study.

Choices of tuning parameters can be crucial. For Stability Selection, Zou [2010] points out that there is no established lower bound for the expected number of true positives, and the tuning parameters π_{thr} and $\tilde{V}$ have significant influences on the true positive rate. They also found in their simulation study that the

number of false positives is usually smaller than the specified $\tilde{V}$. This suggests that less stringent $\tilde{V}$ can help improve signal detection without sacrificing false positive control too much, thus providing a better balance between the two. This is reflected in our results.

Besides penalty level, some methods have additional tuning parameters, such as α in Elastic Net and a in SCAD. They can be tuned together with the penalty level, through e.g. cross-validation, although the multiple-dimensional tuning can be even less stable. However, how to jointly find the optimal choice of multiple tuning parameters is an interesting yet challenging task in practice.

We explicitly defined the true model in terms of exact sparsity (i.e. some coefficients being precisely zero). Although this is the best studied case, in practice such a notion of sparsity may not be realistic and a more reasonable assumption may be that there are a few strong signals, several moderate signals and even more weak signals, but the majority of variables are irrelevant with small, but sometimes non-zero coefficients. In this case, since it may not be possible to find all relevant variables, a good method might be expected to detect all strong and moderate signals while removing the weaker ones. In this vein, Zhang and Huang [2008] consider the problem where weak signals exist outside the ideal model, such that their total signal strength is below a certain level. The authors prove that the Lasso estimate has model size of the correct order, and the selection bias is controlled by the weak signal coefficients and a threshold bias.

In order to make the scope of study manageable, we varied factors in a systematic way so that the relative performance of methods can be easily explained by the features of data. We focused on understanding the variability of performance in a broadly favourable setting. Extending this systematic empirical approach to (the huge range of) less favourable settings, spanning many kinds of model misspecification, could be illuminating, but experimental design would be non-trivial. For example, model can be misspecified in practice, such that measurement error is added to the variables; some relevant variables are not included in the data; variables of higher order or interactions among variables may be missed from the model; a small set of relevant variables may have non-linear relationship with the response. In these cases, it is of interest to see how robust different methods are against the misspecification. For example, Bühlmann and van de Geer [2015] con-

sider high-dimensional inference when model is misspecified. They adapt Lasso to improve its robustness against model misspecification, by hypothesis testing and constructing confidence interval for each coefficient.

Chapter 4

Structural randomised selection

In this chapter we propose an ensemble learning methods to improve the performance of penalised linear regression methods. The method is called STructural RANDomised Selection (STRANDS), which can be combined with sparse regression methods to improve their selection and prediction performance.

4.1 Introduction

The past decades have witnessed the rapid growth of high-dimensional data, such as data from high-throughput technologies and social networks. As previously seen in Chapter 2, high dimensionality and complex correlation structure pose challenges to existing penalised regression methods. In practice, in such settings there is typically no definitive evidence for a single optimal model and evidence is distributed among different candidate models. Ensemble learning (see Section 1.6) is a powerful tool in this situation, where each candidate model serves as a weak learner, and combining them in a strategic way yields a stronger model than each individual model. Ensemble learning methods, such as bagging, can also help decrease the variance of prediction and stabilise the results.

Reducing the dimensionality of data before applying variable selection is a practical way to enhance regression method performance as well as computational efficiency. Popular dimension reduction techniques include principle component analysis [Van Der Maaten et al., 2009] and singular value decomposition [Wang

and Zhu, 2017]. Another approach to reduce dimensionality is through screening, where irrelevant variables are eliminated according to their relationship with the response. For example, sure independence screening [Fan and Lv, 2008] is a straightforward screening approach, where only variables with large univariate regression coefficients are retained in the model before applying a variable selection method.

In Section 1.6.3 we introduced the Random Lasso method [Wang et al., 2011]; this can be viewed as both an ensemble learning approach and a screening method. Recall that the Random Lasso consists of two steps. The first step repeatedly selects q_1 variables randomly from p variables and applies Lasso, and then uses the results to compute an importance measure for each variable; the second step then repeatedly screens out variables based on the importance measures (leaving q_2 variables), and applies Lasso. Final coefficient estimates are then computed. So the first step explores the relationship between each variable and the response, and the second step aims to eliminate variables with no or weak association with the response, based on results of the first step, prior to applying Lasso. The random Lasso approach also bears some resemblance to the random forest ensemble learning method (see Section 1.6) in that in both steps, Lasso is run multiple times on bootstrapped data with only a subset of the p variables, and results are averaged. See Section 1.6.3 for full details of the Random Lasso algorithm.

Random Lasso manages to solve some issues of Lasso, namely, it can simultaneously select highly correlated variables even if they have coefficients of different signs, and the number of selected variables is not limited by sample size. In spite of the nice properties, it has several potential drawbacks:

1. since the search for optimal combination of q_1 and q_2 is achieved by grid search, the method is rather time consuming, particularly for high-dimensional data;

2. fixing q_1 may not be the best way to fully explore the candidate models in Step 1;

3. the threshold $1/n$ on coefficient estimates is somewhat arbitrary since magnitudes of coefficients do not depend on sample size;

4. each bootstrapped sample loses partial information of the original data.

5. the approach does not take account of correlation structure among variables during data splitting; it is known that understanding the correlation structure before regression could help better distinguish relevant variables among correlated irrelevant ones (see e.g. Bühlmann et al. [2013]).

In this section we propose a new modelling strategy, STructural RANDomised Selection (STRANDS), in a similar spirit as Random Lasso. The method takes account of correlation structure of the data, to explore the model space in a structured way. The method has fewer tuning parameters than Random Lasso, and so is more computationally efficient. It uses full sample information, and its selection rule is based on selection probabilities rather than sample size.

The remainder of the section is organised as follows. In Section 4.2 we describe and motivate the proposed method. In Section 4.3 we show results comparing STRANDS with Random Lasso and other popular Lasso variants using both low-dimensional and high-dimensional datasets. In Section 4.4 we investigate the effects of different components of STRANDS on method performance. In Section 4.5 we show results using real data. We conclude with a discussion in Section 4.6.

4.2 Method

STRANDS consists of three steps. It first captures the correlation structure of data. Then it randomly selects variables informed by the correlation structure, before performing regression on those variables. With the information from this model exploration step, the third step aims to randomly remove irrelevant variables while retaining the relevant ones, before performing regression again. The results are averaged across candidate models and the final thresholding rule is based on selection probabilities. STRANDS can be combined with any sparse regression method. We refer to this method as the " base learner ". Each step is explained in more detail below and illustrated in Figures 4.1 and 4.2.

In Step 0, correlation structure is determined. We first apply the sparse regression base learner to the complete data. Then for each of the selected variables, we

Algorithm Correlation clustering algorithm

Input: n by p design matrix $\mathbf{X}$, response vector $\mathbf{Y}$, sparse regression base learner algorithm $\mathcal{A}$ and correlation threshold $\rho_0 > 0$

Output: An independent group of variables G_0 and K correlated groups of variables $G_1 \dots G_K$

Initialise $k = 1$; correlated variables set $G = \emptyset$; remaining variables set $R = \{\mathbf{x_1} \dots \mathbf{x_p}\}$

Run base learner algorithm $\mathcal{A}$ on the whole data.
Let the set of selected variables be $\mathbf{S}$
for $\boldsymbol{x} \in \mathbf{S}$ **do**

    if $\boldsymbol{x} \in G$ **then**
        next
    end if
    $G_k = \{\boldsymbol{x}\}$; $\rho_M = 1$
    while $\rho_M \geq \rho_0$ **do**
        Find the variable $\boldsymbol{x_r} \in R \setminus G_k$ that has the highest median absolute correlation with elements of G_k
        Set ρ_M to be the associated median absolute correlated value
        if $\rho_M \geq \rho_0$ **then**
            $G_k = G_k \cup \{\boldsymbol{x_r}\}$
        end if
    end while
    if $|G_k| \geq 2$ **then**
        $k = k + 1$
        $R = R \setminus G_k$
    end if
end for
$G_0 = R$, $K = k - 1$

Algorithm STructural RANDomised Selection (STRANDS)

Input: n by p design matrix $\mathbf{X}$, response vector $\mathbf{Y}$, sparse regression base learner algorithm $\mathcal{A}$, correlation threshold $\rho_0 > 0$, number of iterations B and threshold probability π_{thr}

Output: coefficient estimate ($\hat{\beta}_j$) and selection probability ($\hat{\pi}_j$) of each variable, and a set of selected variables $\hat{\omega}$

Step 0: Apply correlation clustering algorithm, resulting in one independent group ($\boldsymbol{G_0}$) and K groups of correlated variables ($\boldsymbol{G_1} \ldots \boldsymbol{G_K}$)

Step 1: Calculate importance measure
1a
for $b_1 = 1, \ldots, B$ **do**
(i)
for $k = 0, \ldots, K$ **do**
Sample without replacement from $\boldsymbol{G_k}$ to obtain $\boldsymbol{G'_k} \subseteq \boldsymbol{G_k}$, where $|\boldsymbol{G'_k}|$ is uniformly drawn from $\{0, 1, \ldots, |\boldsymbol{G_k}|\}$
end for
$\boldsymbol{S^{(b_1)}} = \bigcup_{k=0}^{K} \boldsymbol{G'_k}$.
(ii) Apply base learner algorithm $\mathcal{A}$ to $(\boldsymbol{S^{(b_1)}}, \mathbf{Y})$, tuned on the whole range of penalty levels (as *per* default choice of $\mathcal{A}$).
For $\boldsymbol{x_j} \in \boldsymbol{S^{(b_1)}}$, denote the resulting coefficient estimate by $\hat{\beta}_j^{(b_1)}$.
For $\boldsymbol{x_j} \notin \boldsymbol{S^{(b_1)}}$, set $\hat{\beta}_j^{(b_1)} = 0$
end for
1b
for $j = 1, \ldots, p$ **do**

$$m_j = \sum_{b_1=1}^{B} \mathbb{1}\{\boldsymbol{x_j} \in \boldsymbol{S^{(b_1)}}\} \qquad \text{(number of times } \boldsymbol{x_j} \text{ is chosen)}$$

$$\alpha_j = \frac{\sum_{b_1=1}^{B} |\hat{\beta}_j^{(b_1)}|}{m_j} \qquad \text{(mean } |\hat{\beta}_j|)$$

$$\theta_j = \frac{\sum_{b_1=1}^{B} \mathbb{1}(\hat{\beta}_j{}^{(b_1)} \neq 0)}{m_j} \qquad \text{(probability of being selected)}$$

end for

Step 2: Select variables

2a

for $b_2 = 1, \ldots, B$ **do**

(i) Randomly select $\tilde{s} = \lceil \sum_{j=1}^{p} \theta_j \rceil$ ($\lceil \cdot \rceil$ denotes the ceiling function) of the p variables, with selection probability of $\boldsymbol{x_j}$ proportional to $\alpha_j \theta_j$, $j = 1, \ldots, p$. Denote the set of selected variables by $\boldsymbol{S^{(b_2)}}$.

(ii) Apply base learner algorithm $\mathcal{A}$ to $(\boldsymbol{S^{(b_2)}}, \mathbf{Y})$, tuned on the set of optimal penalty levels obtained from Step 0 and 1.

For $\boldsymbol{x_j} \in \boldsymbol{S^{(b_2)}}$, denote the resulting coefficient estimate by $\hat{\beta}_j^{(b_2)}$.

For $\boldsymbol{x_j} \notin \boldsymbol{S^{(b_2)}}$, set $\hat{\beta}_j^{(b_2)} = 0$

end for

2b

for $j = 1, \ldots, p$ **do**

$$\hat{\beta}_j = \frac{\sum_{b_2=1}^{B} \hat{\beta}_j^{(b_2)}}{B}$$

$$\hat{\pi}_j = \frac{\sum_{b_2=1}^{B} I(\hat{\beta}_j^{(b_2)} \neq 0)}{B}$$

end for

Let $\hat{s}_0 = \sum_j I(\hat{\pi}_j \geq \pi_{thr})$

Select variables with the top $\hat{s}_0$ largest coefficient estimates $\hat{\beta}_j$ or the top $\hat{s}_0$ largest selection probabilities $\hat{\pi}_j$, denote the set of selected variables by $\hat{\omega}$

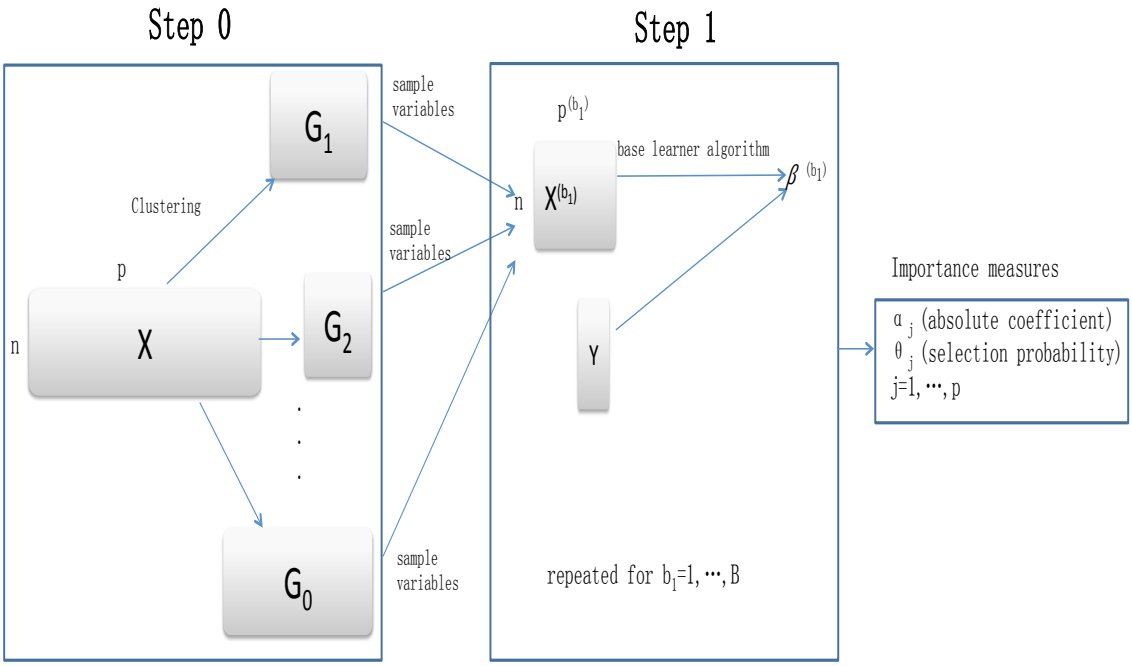


Figure 4.1: An illustration of Step 0 and Step 1 in STRANDS

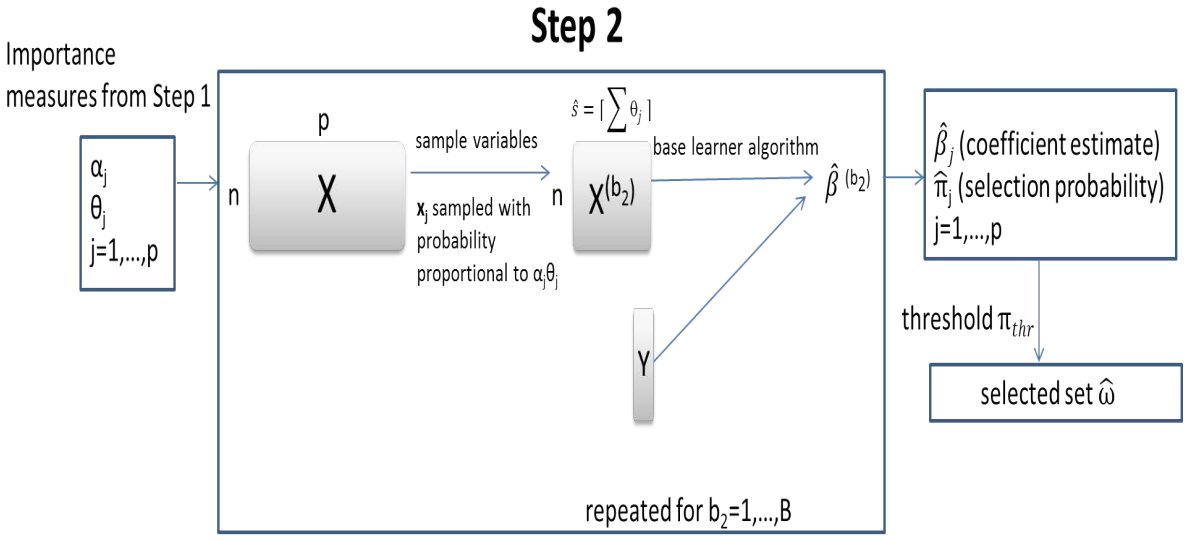


Figure 4.2: An illustration of Step 2 in STRANDS

form a correlated group by iteratively adding in the variable that is most correlated with existing group members. Specifically, for each variable not in the group,

we calculate the median absolute correlation between the variable and the groups members. Then we add in the variable that has the highest correlation. The procedure stops when the largest median absolute correlation is below a threshold. Any variables that do not belong to any of these correlated groups form an " independent group " (note that this means that variables in the independent group are not correlated with any of the correlated groups, but does not necessarily mean that variables in the independent group are uncorrelated with each other). As a result, the p variables are divided into K groups of correlated variables $(G_1, G_2 \dots G_K)$, and an independent group G_0 (see Figure 4.1). These groups are used in the next step to inform sampling of variables.

In Step 1, variables are sampled from each group independently. This helps better explore the model space, as shown in Section 4.4.1. After repeatedly applying the sparse base learner to sampled variables, an average coefficient measure α_j and a selection probability θ_j are obtained for each variable (see Figure 4.1). They are two importance measures quantifying the association between a variable and the response.

In Step 2, sampling of variables is informed by importance measures from Step 1. In particular, the selection probability of $\boldsymbol{x_j}$ is proportional to $\alpha_j\theta_j$. The reason for taking both measures into account rather than just using one of them is that both selection probability and magnitude of coefficient estimate indicate the importance of a variable, and one of the measures can be large, while the other is small. The sparse base learner is then applied to the sampled variables. Variable sampling and sparse regression are repeated multiple times before calculating average coefficient estimate $\hat{\beta}_j$ and selection probability $\hat{\pi}_j$. Variables with large selection probabilities or coefficient estimates are declared as relevant (see Figure 4.2). Similar as Random Lasso, since correlated variables are separated and can be selected from different candidate models, STRANDS also has group selection property, where strongly correlated variables tend to be selected simultaneously.

In Step 1 and 2, when sampling variables, instead of tuning the number of sampled variables as in Random Lasso, they are chosen automatically in STRANDS: in Step 1 a random number of variables are drawn from each of the $K + 1$ groups that result from the clustering algorithm, and in Step 2 the number of selected variables is the sum of selection probabilities from Step 1, which is the expected

model size based on evidence from model space exploration. The estimate is informative and robust. Selection probability of a strong signal in Step 1 will be close to one; selection probabilities among correlated variables in Step 1 may randomly distribute, but their sum is a robust estimate of total number of signals among the correlated variables. The automatic choices of parameters make STRANDS much more computationally efficient than Random Lasso. Unlike Random Lasso, STRANDS does not use bootstrapped samples, and take advantage of more sample information during model exploration and analysis. We propose a thresholding rule not dependent on sample size n (Random Lasso uses a threshold $1/n$), but based on selection probabilities, e.g., variables selected in more than a fraction of sub-models in Step 2 are declared as important; we also keep variables with largest coefficient estimates. Since the selection probabilities of signals tend to be pushed towards 1, while the non-signals towards 0, a threshold of $\pi_{thr} = 0.5$ is reasonable, and it seems to work well in practice. In general we recommend B to be a fairly large value (e.g. $B \geq 200$) for effective exploration of model space and stable results.

Note that in Step 1, we use the grid of tuning parameters provided by the R packages by default, to find the best penalty level for each sub-model (by e.g. cross-validation). However, in Step 2, using a grid that ignores the existence of previous steps may not be appropriate. Specifically, Zhu and Yang [2014] prove that, after screening, one should apply the penalty level optimal for the full data on the survived variables, in order to achieve unbiased estimation. In other words, if we treat the survived variables after screening as if they are given in the first place, the survived variables tend to prefer smaller level of penalty, such that bias can occur. This is a systematic overestimation of effects that are ascertained by screening. Coefficient estimates of survived variables are not shrunk as much as they should be towards zero, if the same samples are used for post-screening analysis. Motivated by this argument, the grid of tuning parameters for datasets in Step 2 is chosen to be the union of all optimal penalty levels tuned based on full data in Step 0 and all datasets in Step 1. The purpose is to mitigate the screening bias, but as we will see in the results, this bias does not altogether disappear.

4.3 Results

4.3.1 Low-to-moderate-dimensional settings

In this section we perform several simulation studies to demonstrate the proposed method and compare to popular variants of Lasso. Data are generated using Model (1.1): $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$. For all simulation settings, each variable follows a standard normal distribution, but the correlation structure may vary. $\epsilon_i \sim N(0, \sigma^2)$ for $i = 1, \ldots, n$. Variables are standardised to have mean zero and variance one, and responses are centred.

Example 4.1 and Example 4.2 were used in the Lasso paper [Tibshirani, 1996], where correlation structures follow the Toeplitz design. Example 4.3 was used in the Random Lasso paper [Wang et al., 2011], where coefficients of highly correlated variables have different signs. We have $p > n$ in Examples 4.4 and 4.5, one with orthogonal variables and the other with pairwise correlation structure. The details of the six examples are as follows:

Example 4.1. There are $p = 8$ variables, and the pairwise correlation between variables x_i and x_j is $0.5^{|i-j|}$, $i, j = 1 \ldots 8$. The true coefficients are $\boldsymbol{\beta} = (3, 1.5, 0, 0, 2, 0, 0, 0)$, $\sigma = 3$, SNR ≈ 1.5.

Example 4.2. The model is the same as in Example 4.1, except $\beta_j = 0.85$ for $j = 1 \ldots 8$. i.e., a non-sparse model. SNR ≈ 1.25.

Example 4.3. There are $p = 40$ variables. The first 10 variables are relevant variables, and the correlation between each pair of them is 0.9. The rest 30 variables are independent from each other and independent from the 10 relevant variables. Relevant variables have coefficients 3, 3, 3, 3, 3, -2, -2, -2, -2, -2. $\sigma = 3$. SNR $\approx$ 1.78.

Example 4.4. There are $p = 300$ variables. All variables are i.i.d. drawn from $N(\mathbf{0}, \mathbf{I})$. 10 relevant variables are randomly allocated among p variables, which

have coefficients 3, 3, 3, 3, 3, 4, 4, 4, 4, 4. $\sigma = 3$, SNR ≈ 3.65.

Example 4.5. Same as in Example 4.4, but the first 100 variables are 10 blocks of 10 pairwise correlated variables with correlation $\rho = 0.7$, with one relevant variable in each block; the ten blocks are independent from each other. The rest 200 variables are independent from each other, and independent from the 10 blocks of correlated variables, with no relevant variables. SNR≈ 3.65.

We compare Lasso, Elastic net (ENet, $\alpha = 0.5$), Adaptive Lasso (AdaLasso, Lasso estimates as initial weights) with Random Lasso (RLasso), STRANDS-Lasso (STRD-Lasso) and STRANDS-Adaptive Lasso (STRD-AdaLasso). In other words, we consider two base learners for STRANDS. Hereafter when referring to STRANDS, we mean both STRANDS-Lasso and STRANDS-AdaLasso. We use $B = 300$, correlation threshold $\rho_0 = 0.5$, $\pi_{thr} = 0.5$ for STRANDS, and for Random Lasso, we use $B = 300$ and search a grid of 0, $0.2p$, $0.4p$, $0.6p$, $0.8p$ and p to find the optimal q_1 and q_2. For all methods penalty level λ is chosen by fivefold cross-validation (note that for Step 2 of STRANDS the range of penalty levels is constrained). It is worth mentioning that Wang et al. [2011] propose to tune q_1 and q_2 using validation data, while in our implementation tuning parameters are determined by cross validation.

To assess selection performance, we use the number of true positives (TP), number of false positives (FP), PPV $= \frac{\text{TP}}{\text{TP+FP}}$. To assess prediction performance, we use mean squared error, defined as MSE$=(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^T \mathbf{V} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})$, where $\mathbf{V}$ is the population covariance matrix of $\mathbf{X}$ (see Section 1.7 for more details). For STRANDS and RLasso, we use the final selected model after thresholding to assess their selection and prediction performance. Results are shown in Tables 4.1 $\sim$ 4.5 and are averages across 100 replicates, with bootstrap standard errors in parentheses. We highlight the method with best performance in bold in each row.

Selection STRD-Lasso has better false positive control than Lasso/ENet in all cases, and always has similar or better true positive rate than Lasso. When $p > n$ (Tables 4.4, 4.5), it also achieves similar true positive rate as ENet. In contrast, RLasso in general has good true positive rates, but with poor false positive control.

n		Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
20	#FP	2.28(0.15)	2.58 (0.16)	1.16 (0.11)	2.97 (0.12)	1.95 (0.12)	**0.99 (0.1)**
	#TP	2.73 (0.05)	2.8 (0.05)	2.39 (0.07)	**2.87 (0.04)**	2.77 (0.05)	2.4 (0.07)
	PPV	0.6 (0.02)	0.57 (0.018)	0.73 (0.021)	0.51 (0.014)	0.62 (0.019)	**0.75 (0.022)**
	MSE	4.45 (0.35)	4.40 (0.35)	4.49 (0.34)	4.16 (0.23)	**3.64 (0.25)**	3.95 (0.26)
50	#FP	2.17 (0.16)	2.72 (0.14)	1.05 (0.12)	3.05 (0.13)	2.02 (0.13)	**0.64 (0.08)**
	#TP	**3 (0)**	**3 (0)**	2.92 (0.03)	**3 (0)**	**3 (0)**	2.96 (0.02)
	PPV	0.64 (0.019)	0.56 (0.015)	0.8 (0.02)	0.52 (0.014)	0.64 (0.018)	**0.86 (0.016)**
	MSE	1.37 (0.09)	1.40 (0.09)	1.32 (0.12)	1.32 (0.09)	1.18 (0.08)	**1.09 (0.08)**

Table 4.1: Example 4.1 results: $p = 8, s_0 = 3, \sigma = 3, \rho(x_i, x_j) = 0.5^{|i-j|}, \boldsymbol{\beta} = (3, 1.5, 0, 0, 2, 0, 0, 0)$.

n		Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
20	#TP	5.39 (0.18)	5.84 (0.16)	4.06 (0.15)	**5.85 (0.13)**	5.43 (0.14)	3.95 (0.13)
	MSE	4.95 (0.32)	4.55 (0.31)	5.53 (0.3)	4.80 (0.19)	**4.17 (0.23)**	5.09 (0.28)
50	#TP	7.44 (0.07)	**7.54 (0.06)**	6.09 (0.12)	7.39 (0.07)	7.44 (0.07)	5.89 (0.11)
	MSE	1.41 (0.07)	**1.32 (0.06)**	2.02 (0.10)	2.13 (0.09)	1.38 (0.06)	2.04 (0.09)

Table 4.2: Example 4.2 results: Same as Table 4.1, $\beta_j = 0.85\ \forall j$.

n		Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
50	#FP	4.22 (0.49)	5.42 (0.54)	2.68 (0.33)	12.91 (0.36)	3.9 (0.27)	**2.54 (0.2)**
	#TP	3.24 (0.11)	4.87 (0.11)	2.51 (0.1)	6.6 (0.17)	**6.82 (0.19)**	4.27 (0.18)
	PPV	0.59 (0.028)	0.57 (0.023)	0.65 (0.030)	0.35 (0.008)	0.66 (0.019)	**0.67 (0.021)**
	MSE	6.32 (0.19)	6.36 (0.25)	7.22 (0.23)	6.16 (0.16)	**5.22 (0.20)**	6.17 (0.19)
100	#FP	10.71 (1.08)	10.68 (1.01)	5.29 (0.48)	16.23 (0.33)	5.34 (0.41)	**2.75 (0.24)**
	#TP	5.96 (0.27)	6.4 (0.23)	5.16 (0.28)	9.32 (0.08)	**9.61 (0.08)**	8.47 (0.13)
	PPV	0.53 (0.026)	0.53 (0.026)	0.62 (0.025)	0.37 (0.004)	0.69 (0.017)	**0.79 (0.014)**
	MSE	4.92 (0.11)	4.89 (0.097)	4.81 (0.15)	3.18 (0.12)	**2.21 (0.10)**	2.47 (0.13)

Table 4.3: Example 4.3 results: $p = 40, s_0 = 10, \sigma = 3$. One block of 10 pairwise correlated variables (correlation $\rho = 0.9$), all of which are signals, with coefficients 3, 3, 3, 3, 3, -2, -2, -2, -2, -2. The other 30 variables are independent from each other, and independent from the 10 correlated variables, all of which are noise variables.

n		Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
50	#FP	18.88 (1.28)	25.95 (1.43)	10.36 (0.61)	19.28 (0.67)	15.08 (0.6)	**8.36 (0.4)**
	#TP	6.98 (0.27)	7.54 (0.25)	6.44 (0.28)	7.62 (0.15)	**7.64 (0.18)**	7.09 (0.21)
	PPV	0.34 (0.019)	0.28 (0.017)	0.42 (0.017)	0.3 (0.010)	0.36 (0.012)	**0.48 (0.016)**
	MSE	72.39 (3.25)	74.96 (2.67)	69.13 (3.80)	67.53 (2.43)	61.91 (3.01)	**60.34 (3.0)**
100	#FP	38.92 (1.4)	50.79 (1.55)	16.42 (0.67)	47.62 (0.72)	23.71 (0.57)	**10.12 (0.36)**
	#TP	10 (0)	10 (0)	10 (0)	10 (0)	10 (0)	10 (0)
	PPV	0.22 (0.0062)	0.18 (0.0046)	0.41 (0.013)	0.18 (0.0023)	0.31 (0.0055)	**0.52 (0.011)**
	MSE	8.11 (0.33)	11.32 (0.43)	4.87 (0.25)	**3.65 (0.17)**	7.39 (0.31)	3.72 (0.12)

Table 4.4: Example 4.4 results: $p = 300, s_0 = 10, \sigma = 3$. Variables are i.i.d. drawn from $N(\mathbf{0}, \mathbf{I})$. 10 signals are randomly allocated among p variables, with coefficients 3, 3, 3, 3, 3, 4, 4, 4, 4, 4.

n		Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
50	#FP	29.42 (1.17)	38.6 (1.01)	15.86 (0.52)	26.8 (0.73)	23.49 (0.55)	**12.64 (0.39)**
	#TP	6.6 (0.21)	**7.33 (0.17)**	5.8 (0.22)	6.45 (0.17)	7.32 (0.17)	6.34 (0.19)
	PPV	0.19 (0.0066)	0.17 (0.0046)	0.27 (0.009)	0.2 (0.0067)	0.24 (0.0063)	**0.34 (0.012)**
	MSE	60.32 (2.20)	58.90 (1.74)	60.47 (2.44)	64.86 (1.95)	**51.18 (1.72)**	54.21 (2.26)
100	#FP	41.5 (1.41)	56.94 (1.42)	18.09 (0.62)	43.62 (0.99)	35.13 (0.61)	**13.41 (0.5)**
	#TP	**9.99 (0.01)**	9.98 (0.01)	9.97 (0.02)	9.98 (0.01)	9.98 (0.01)	9.96 (0.02)
	PPV	0.21 (0.005)	0.16 (0.003)	0.38 (0.01)	0.19 (0.004)	0.23 (0.003)	**0.45 (0.01)**
	MSE	9.67 (0.44)	13.49 (0.54)	5.90 (0.31)	**5.26 (0.44)**	9.94 (0.42)	5.85 (0.39)

Table 4.5: Example 4.5 results: $p = 300, s_0 = 10, \sigma = 3$, 10 blocks of 10 pairwise correlated variables (correlation $\rho = 0.7$), each having one signal. The other 200 variables are independent from each other, and independent from the 10 blocks of correlated variables, all of which are noise variables. Signals have coefficients 3, 3, 3, 3, 3, 4, 4, 4, 4, 4.

It selects more false positives than STRD-Lasso in all cases. Example 4.3 is the motivating setting for Random Lasso where coefficients of highly correlated variables have different signs. Thanks to group selection property, both STRD-Lasso and RLasso succeed in detecting significantly more correlated relevant variables than other methods, but STRD-Lasso has much better false positive control and higher power than Random Lasso.

When $p > n$ and variables are orthogonal to each other (Table 4.4), the advantage of false positive control of STRD-Lasso compared to Lasso, ENet and RLasso is more obvious than in the block correlation design (Table 4.5). One possible reason is that in Example 4.5, only one out of 10 correlated variables is relevant in each block, but STRANDS tends to simultaneously select multiple correlated variables, which introduces false positives.

Comparative performance of AdaLasso and STRD-AdaLasso is broadly similar to that of Lasso and STRD-Lasso. AdaLasso typically has good false positive control, and such advantage is obvious when $p > n$ (Tables 4.4 and 4.5). STRD-AdaLasso typically improves AdaLasso on power and false positive control, and this is especially true in Example 4.3 where relevant variables with different signs are highly correlated.

Prediction STRANDS achieves similar or better MSEs than its base learner in all cases. For the smaller sample sizes ($n = 20$ for Examples 4.1, 4.2 and $n = 50$ for Examples 4.3–4.5), STRD-Lasso's prediction accuracy is notably better than other methods. In "easier" settings where sample size is larger ($n = 50$ for Examples 4.1, 4.2 and $n = 100$ for Examples 4.3–4.5), its advantage in prediction is typically less, and can be outperformed by RLasso and AdaLasso when $p > n$ (Examples 4.4 and 4.5).

Example 4.2 is the motivating setup for Elastic Net where the underlying model is not sparse, yet STRD-Lasso manages to achieve comparable prediction error to Elastic Net. Similar to RLasso, STRD-Lasso is favourable when relevant variables with different signs are highly correlated (Example 4.3). Here, its prediction error is significantly better than Lasso, ENet, and also RLasso (for example, when $n = 100$, MSEs of STRD-Lasso and RLasso are 2.21 and 3.18 respectively). The predictive performance of Random Lasso is more variable, and seems to be competitive when

$p > n$ and n is large (see $n = 100$ for Examples 4.4, 4.5).

Comparative performance of AdaLasso and STRD-AdaLasso is broadly similar to that of Lasso and STRD-Lasso. AdaLasso has inferior performance compared to Lasso and ENet when strong multicollinearity exists (see e.g. Example 4.3), and is competitive for larger n. STRD-AdaLasso achieves similar or better predictive performance compared to AdaLasso in all cases, and the improvement is the most significant in Example 4.3 (when $n = 100$, the MSEs of AdaLasso and STRD-AdaLasso are 4.81 and 2.47 respectively).

Null model We further compare method performance in a null model, to assess false positive control and bias control. Specifically, we generate data where 300 variables and a response are i.i.d. drawn from $N(\mathbf{0}, \mathbf{I})$, so there is no linear relationship between response and any variables. The results are summarised in Table 4.6.

n		Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
50	#FP	4.53 (0.8)	6.3 (1.07)	**2.99 (0.51)**	20.92 (0.4)	5.91 (0.5)	4.41 (0.31)
	MSE	0.034 (0.007)	**0.031 (0.006)**	0.131 (0.018)	0.285 (0.01)	0.216 (0.013)	0.253 (0.013)
100	#FP	3.07 (0.63)	3.41 (0.68)	**2.08 (0.48)**	38.76 (0.61)	4.73 (0.47)	3.49 (0.39)
	MSE	0.011 (0.002)	**0.009** (0.002)	0.059 (0.012)	0.28 (0.01)	0.11 (0.01)	0.13 (0.011)

Table 4.6: Example 4.6 results: $p = 300$, all variables and response are i.i.d. drawn from $N(\mathbf{0}, \mathbf{I})$

Since the screening in STRANDS introduces bias, its has poorer false positive control than its base learner, and prediction error is also inflated, but less severe than RLasso. Since the thresholding rule $1/n$ is somewhat arbitrary, the false positive control of RLasso is rather poor in this case. AdaLasso has the best false positive control, but its prediction is less competitive than Lasso and ENet, since the AdaLasso estimate is biased further away from its initial weights, the Lasso estimate.

4.3.2 High-dimensional settings

In this section we apply STRANDS and its competitors to some high-dimensional datasets, to evaluate their comparative performance.

Datasets are generated as in Section 2.2.1. Recall that s_0 is the total number of relevant variables, ρ is the correlation among correlated variables in a block, p^B is the block size, s_0^B is the number of relevant variables in a block, and $r = n/(s_0 log(p - s_0))$ is a measure of scenario difficulty. We consider the following scenarios with different designs and features:

Example 4.7: Independence design, $n = 200, p = 500, s_0 = 20$, SNR=1, $r = 1.62$.
Example 4.8: Pairwise correlation design, $n = 100, p = 500, s_0 = 20$, SNR=4, $\rho = 0.9, p^B = 10, s_0^B = 5$, $r = 0.81$.
Example 4.9: Pairwise correlation design, $n = 300, p = 1000, s_0 = 20$, SNR=2, $\rho = 0.7, p^B = 100, s_0^B = 2$, $r = 2.18$.
Example 4.10: As Example 4.9, but with Toeplitz correlation design as described in Section 2.2.1.

In Example 4.7 the challenge is high noise level. In Example 4.8 strong multicollinearity exists with small r, and from the comparison study in Chapter 2 we see that large ρ and s_0^B, and small p^B and r tend to favour l_2 penalty, particularly for selection. In Examples 4.9 and 4.10 p^B is large, and the challenge is to find the relevant variables among many correlated irrelevant ones. The only difference is in the correlation structure (pairwise versus Toeplitz). Parameters are tuned or set as before, and we report #TP, #FP, PPV and MSE of each method. The results are summarised in Tables 4.7 – 4.10.

Selection Across high-dimensional settings, RLasso's true positive rate is among the best, but it has the worst false positive control. This is mainly because the thresholding rule $1/n$ can be too lenient when n is large. In Examples 4.7 and 4.8, STRD-Lasso has better false positive control than Lasso and ENet, with similar or better true positive rates. In Example 4.8 where multiple correlated variables are relevant, consistent with findings in systematic comparison study in Chapter 2, ENet has significant gain in TPR over Lasso, and STRD-Lasso also manages to improve upon the true positive rate of Lasso. In Example 4.9 and 4.10, large p^B makes it very difficult to distinguish between correlated variables. Lasso and ENet select many false positives, and STRD-Lasso has similar or slightly worse

performance than Lasso with respect to true positive and false positive. The tendency of simultaneously selecting correlated variables may be the reason why STRD-Lasso selects slightly more variables than Lasso in the pairwise correlation design.

Comparative performance of AdaLasso and STRD-AdaLasso is broadly similar to that of Lasso and STRD-Lasso. AdaLasso typically has good false positive control but less competitive true positive rates. STRD-AdaLasso has similar true positve rates as AdaLasso, and improves false positive control of AdaLasso except in Example 4.9.

Overall, we see that STRANDS has less benefit in high-dimensional settings with large correlated blocks and few signals, compared to low-to-moderate dimensional settings.

Prediction Lasso and ENet have similar predictive performance except in Example 4.8, where relevant variables are strongly correlated, and ENet outperforms Lasso. As expected, both RLasso and STRD-Lasso improves Lasso's prediction accuracy in Example 4.8, and STRD-Lasso has marginally the best performance. In other three examples, STRD-Lasso has similar or slightly worse predictive performance than Lasso, and the comparative performance of STRD-Lasso and RLasso varies. RLasso is noticeably worse than STRD-Lasso in the independence design (Example 4.7, MSE for STRD-Lasso and RLasso are 115.36 and 122.79). In moderate pairwise correlation design (Example 4.9), they have similar performance, while in Toeplitz design RLasso achieves the best prediction accuracy among all methods (Example 4.10).

AdaLasso's predictive performance is among the worst in all settings. This is probably because the initial weights obtained by Lasso are not accurate due to high noise level or strong multicollinearity. STRD-Adalasso has similar or better performance than AdaLasso, and the improvements are significant in data with high noise level (Example 4.7, MSE for AdaLasso and STRD-AdaLasso are 149.57 and 128.91) or strong multicollinearity (Example 4.8, MSE for AdaLasso and STRD-AdaLasso are 32.38 and 27.34). However, even with the improvements, STRD-AdaLasso still does not outperform other methods in those settings.

Similar to selection performance, we see that the benefit of STRANDS in high-

dimensional settings is not as clear as in low-to-moderate dimensional settings.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
#FP	37.36 (2.32)	44.12 (2.62)	31.31 (2.04)	120.23 (1.72)	30.27 (1.33)	**25.42 (1.07)**
#TP	14.69 (0.38)	15.2 (0.37)	14.11 (0.49)	**18.16 (0.14)**	14.73 (0.31)	14.36 (0.29)
PPV	0.32 (0.015)	0.29 (0.013)	0.37 (0.022)	0.13 (0.0019)	0.34 (0.0095)	**0.38 (0.011)**
MSE	**111.24 (2.64)**	112.43 (2.40)	149.57 (4.01)	122.79 (2.85)	115.36 (2.89)	128.91 (3.4)

Table 4.7: Example 4.7: Independence design, $n = 200, p = 500, s_0 = 20$, SNR=1.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
#FP	20.12 (1.03)	24.56 (1.03)	11.34 (0.65)	37.19 (1.09)	15.72 (0.6)	**6.62 (0.37)**
#TP	15.58 (0.22)	17.75 (0.19)	12.23 (0.22)	**18.25 (0.14)**	16.08 (0.22)	12.17 (0.23)
PPV	0.46 (0.015)	0.44 (0.011)	0.54 (0.016)	0.34 (0.0076)	0.52 (0.012)	**0.66 (0.014)**
MSE	28.80 (1.02)	25.30 (0.88)	32.38 (1.49)	24.87 (0.84)	**24.31 (1.04)**	27.34 (0.94)

Table 4.8: Example 4.8: Pairwise correlation design, $n = 100, p = 500, s_0 = 20$, SNR=4, $\rho = 0.9, p^B = 10, s_0^B = 5$.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
#FP	73.58 (1.49)	84.53 (1.74)	**36.2 (1.53)**	129 (3.59)	79.81 (1.23)	39.56 (0.96)
#TP	18 (0.16)	18.23 (0.14)	16.27 (0.23)	**18.64 (0.1)**	17.91 (0.15)	16.3 (0.24)
PPV	0.2 (0.0034)	0.18 (0.003)	**0.33 (0.0097)**	0.13 (0.0034)	0.19 (0.003)	0.3 (0.0071)
MSE	**33.14** (0.76)	33.94 (0.73)	38.02 (1.25)	33.98 (0.86)	34.79 (0.74)	39.35 (1.07)

Table 4.9: Example 4.9: Pairwise correlation design, $n = 300, p = 1000, s_0 = 20$, SNR=2, $\rho = 0.7, p^B = 100, s_0^B = 2$.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
#FP	58.59 (1.16)	75.44 (1.42)	33.06 (1.4)	93.16 (2.48)	58.73 (1.06)	**28.81 (0.82)**
#TP	15.36 (0.22)	**16.97 (0.22)**	12.7 (0.24)	16.92 (0.23)	15.3 (0.24)	12.88 (0.24)
PPV	0.21 (0.0041)	0.19 (0.0032)	0.29 (0.0082)	0.16 (0.0033)	0.21 (0.0039)	**0.32 (0.0089)**
MSE	23.54 (0.53)	23.57 (0.51)	26.66 (1.07)	**21.73 (0.55)**	24.55 (0.73)	25.75 (0.75)

Table 4.10: Example 4.10: As Example 4.9, but with Toeplitz correlation design as described in Section 2.2.1.

The reason STRANDS is less competitive in high dimensional settings may be that a) the model space is huge, and sub-models in Step 1 may not properly represent the model space, such that the importance measures obtained from Step 1 is not as accurate as in low dimensional settings; b) the screening bias is more severe as only a small proportion of variables are retained after screening.

4.4 Further analysis of STRANDS

4.4.1 Structural subsampling

To illustrate the advantage of clustering based on correlation, we present the results of two alternative implementations of STRANDS: a) Random Clustering (RC), where the number of clusters $K+1$ and cluster sizes $|G_0|\dots|G_K|$ are the same as in STRANDS, but variables are randomly assigned to each cluster, and b) No Clustering (NC), where all variables are treated as one group. We compare these implementations with the original STRANDS approach (with clustering based on correlation) and with Random Lasso, for Examples 4.3 and 4.5, which have pairwise correlation structures. Results are shown in Tables 4.11 and 4.12.

n		STRD-Lasso	STRD-Lasso (RC)	STRD-Lasso (NC)	RLasso
50	#FP	**3.9 (0.27)**	4.89 (0.32)	4.32 (0.33)	12.91 (0.36)
	#TP	**6.82 (0.19)**	4.7 (0.17)	4.89 (0.15)	6.6 (0.17)
	PPV	**0.66 (0.019)**	0.53 (0.019)	0.58 (0.02)	0.35 (0.008)
	MSE	**5.22 (0.20)**	6.68 (0.19)	6.62 (0.20)	6.16 (0.16)
100	#FP	**5.34 (0.41)**	6.34 (0.45)	7.03 (0.52)	16.23 (0.33)
	#TP	**9.61 (0.08)**	8.77 (0.13)	8.86 (0.14)	9.32 (0.08)
	PPV	**0.69 (0.017)**	0.63 (0.017)	0.62 (0.017)	0.37 (0.004)
	MSE	**2.21 (0.10)**	3.18 (0.13)	3.29 (0.14)	3.18 (0.12)

Table 4.11: Performance of STRANDS with random clustering (RC) and no clustering (NC) for Example 4.3. Results of RLasso and STRD-Lasso are the same as in Table 4.3.

For data with clear correlation structure, the clustering step of STRANDS can significantly improve its performance. In Example 4.3, RC and NC both lead to much worse TP, FP , PPV and MSE (Table 4.11), and in Example 4.5 they lead to more FP and inferior MSE, and similar TP, although the deterioration in performance is not as striking.

The purpose of clustering is to explore more efficiently the model space, rather than only exploring models with certain characteristics. This is illustrated in Figure 4.3, using the pairwise correlation structure of Example 4.3. The data has one correlated group of size 10, and one independent group of size 30. Without clustering step, we sample variables from the set of all variables, and the ratio of

n		STRD-Lasso	STRD-Lasso (RC)	STRD-Lasso (NC)	RLasso
50	#FP	**23.49 (0.55)**	28.05 (0.59)	27.5 (0.59)	26.8 (0.73)
	#TP	7.32 (0.17)	**7.39 (0.16)**	7.35 (0.17)	6.45 (0.17)
	PPV	**0.24 (0.0063)**	0.21 (0.006)	0.22 (0.006)	0.2 (0.0067)
	MSE	**51.18 (1.72)**	52.38 (1.74)	52.60 (1.8)	64.86 (1.95)
100	#FP	**35.13 (0.61)**	41.06 (0.74)	40.1 (0.67)	43.62 (0.99)
	#TP	9.98 (0.01)	**9.99 (0.01)**	**9.99 (0.01)**	9.98 (0.01)
	PPV	**0.23 (0.003)**	0.2 (0.003)	0.2 (0.003)	0.19 (0.004)
	MSE	9.94 (0.42)	10.57 (0.43)	10.58 (0.46)	**5.26 (0.44)**

Table 4.12: Performance of STRANDS with random clustering (RC) and no clustering (NC) for Example 4.5. Results of RLasso and STRD-Lasso are the same as in Table 4.5.

numbers of sampled variables from the two groups will be close to the ratio of the corresponding group sizes, i.e., 1:3 in this case. So we are likely to focus on a small regime of model space. With clustering, we sample at random from each group independently, and all combinations of variables from different groups are equally likely, so the exploration in Step 1 of STRANDS is more comprehensive and effective.

4.4.2 Two-step variable sampling

In Step 1 STRANDS samples in a uniform way from each group of variables obtained in the clustering step, and applies the base learner algorithm to the random subsets of variables. Although the averaged results on random subset of variables do not provide accurate coefficient estimates since candidate models can be mis-specified, they offer guidance on relative importance of variables and underlying sparsity of the true model. Step 2 exploits the relative importance by random sampling, such that variables with larger importance scores from Step 1 are more likely to be sampled. Thus relevant variables are more likely to be included in candidate models while irrelevant one are more likely to be excluded. Step 2 evaluates joint effect of relevant variables by applying the base learner algorithm, and calculates importance scores, which are more accurate than those in Step 1. This is illustrated in Figure 4.4. We use Example 4.3 with clear correlation structure, and Lasso as the base learner for STRANDS, to show how

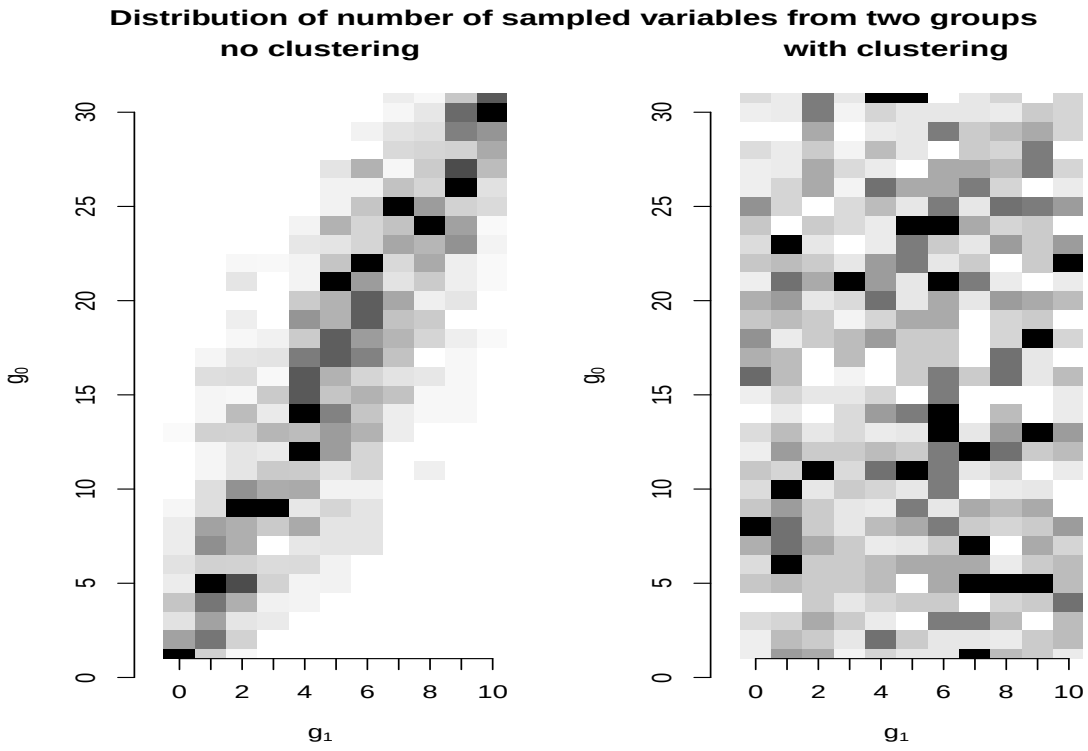


Figure 4.3: Influence of the clustering step (Step 0 of STRANDS), on the sampling of variables (Step 1a (i)) for Example 4.3, where there are two groups. Let G_0 be the group of 30 independent variables and G_1 be the group of 10 highly correlated variables. Left panel shows the result without clustering step (i.e., all variables are treated as one group) and right panel shows the result with clustering step. Sampling of variables is repeated 1000 times, and each time corresponds to a combination of g_0 (number of variable sampled from G_0, y axis) and g_1 (number of variables sampled from G_1, x axis). Darker colour indicates that the combination of g_0 and g_1 appears more frequently than others.

the importance scores change between Step 1 and Step 2. Left panel shows the comparison of selection probabilities of Step 1 and Step 2, and right panel shows the comparison of (absolute) coefficient estimates. We see that selection probabilities for relevant variables are boosted after importance score-guided sampling in Step 2, and selection probabilities for irrelevant variables are typically down-weighted. Similarly, coefficient estimates of relevant variables are also improved after Step 2. This justifies the necessity of Step 2 of STRANDS.

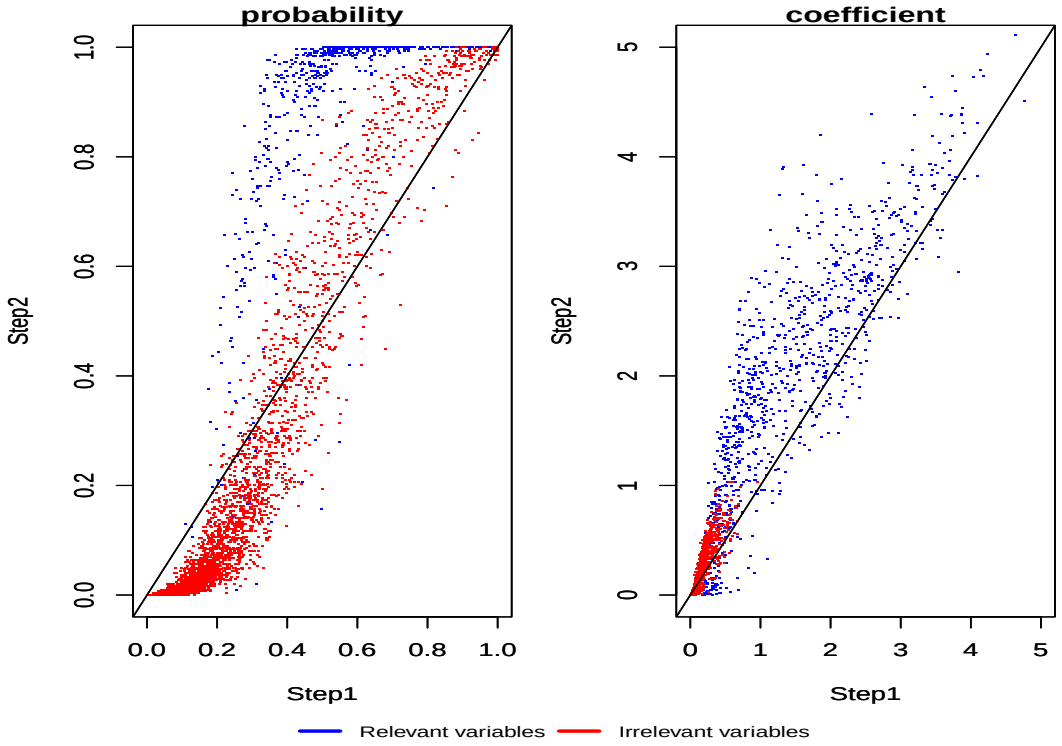


Figure 4.4: Comparison of importance scores in Step 1 and 2 of STRD-Lasso for Example 4.3, as in Table 4.3, $n = 100$. Simulation is repeated 100 times, and each dot corresponds to a score (probability or absolute coefficient) of a variable in one simulation. Left panel shows the comparison of selection probabilities of Step 1 (θ_j) and Step 2 ($\hat{\pi}_j$), and right panel shows the comparison of (absolute) coefficient estimates of Step 1 (α_j) and Step 2 ($|\hat{\beta}_j|$). Relevant variables and irrelevant variables are presented by blue and red dots respectively.

4.5 Application to real data

In this section we analyse three real data examples by using STRANDS and other variants of Lasso. All variables are standardised to have mean zero and variance one. We first run all methods on the full dataset to get model sizes, and then compare their predictive performance as follows: random split the dataset into a training dataset ($\boldsymbol{X_{train}}$, $\boldsymbol{Y_{train}}$) with 90% of samples and a test dataset of the other 10% of samples ($\boldsymbol{X_{test}}$, $\boldsymbol{Y_{test}}$); models are fitted on the training data and the coefficient estimates $\hat{\boldsymbol{\beta}}$ are used to predict the response of the test data,

$$\text{Prediction Error} = ||\boldsymbol{Y_{test}} - \boldsymbol{X_{test}}\hat{\boldsymbol{\beta}}||_2^2/n_{test}. \tag{4.1}$$

We applied Lasso, Elastic Net ($\alpha = 0.5$), Adaptive Lasso, Random Lasso, STRANDS-Lasso, and STRANDS-Adaptive Lasso to fit linear regression models. Parameters and tuning strategies are the same as in simulation study. Data splitting is repeated 100 times, and mean prediction errors across the 100 times are reported, with bootstrap standard errors in parentheses. We also report the computation time of every method for running on the full data. In all examples STRANDS-Lasso takes significantly less computation time than Random Lasso (STRANDS-Lasso is 3-4 times faster than Random Lasso).

4.5.1 Gene expression and aging

The first dataset is from Lu et al. [2004]. The study aims to discover the relationship between aging and gene expression levels in human frontal cortex. The response is ages of $n = 30$ patients (ranging from 26 to 106 years), and explanatory variables are expression levels of 12,625 genes measured by microarray. Linear model is built to find those genes that can predict the age of a patient. Gene expression values are log-transformed, and they filter out genes based on local false non-discovery rates. After preprocessing, $p = 403$ variables are retained as candidate variables. This dataset has previously been analysed in Zuber and Strimmer [2011], and is available in R package *care*. As in Zuber and Strimmer [2011], the response is standardised.

The results of aging data analysis are summarised in Table 4.13. We see that

STRANDS-Lasso achieves the smallest mean prediction error, followed by Elastic Net and Lasso. Prediction errors of Adaptive Lasso and Random Lasso are less satisfactory, perhaps because they both have overly sparse models and missed more signals. For Random Lasso, this could be due to the fact that sample size of training data $n = 27$ is small, so that the coefficient estimate threshold $1/n$ is too restrictive and the resulting model is too conservative. On the other hand, STRANDS-Lasso and STRANDS-Adaptive Lasso have similar model sizes as their base learners, with improved prediction errors respectively.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
Model Size	23	29	9	13	21	10
Prediction Error	0.344 (0.028)	0.313 (0.024)	0.465 (0.042)	0.563 (0.036)	0.291 (0.019)	0.429 (0.031)
Time (s)	<1	<1	<1	205.33	53.24	102.15

Table 4.13: Real data analysis : age data, $n = 30, p = 403$

4.5.2 Gene expression and Bardet-Biedl syndrome

Next we consider the dataset of mammalian eye tissue, reported in Scheetz et al. [2006]. Tissue harvesting is applied for 120 male rats' eyes, followed by microarray analysis. Expression levels of over 28,000 genes are measured. It is known that TRIM32 gene is the cause of Bardet-Biedl syndrome [Chiang et al., 2006], and linear regression is applied to find genes most related to TRIM32. It is believed that only a small number of genes are associated with TRIM32, and after pre-processing, the resulting dataset consists of 200 variables, which is available in R package *flare*.

The results of eye data analysis are summarised in Table 4.14. STRANDS-Lasso again achieves the smallest mean prediction error, followed by Elastic Net and Lasso. Adaptive Lasso is overly sparse and worst for prediction purpose, while Random Lasso does not do well for prediction either, and selects less variables than Lasso. Unlike in age data, STRANDS-Lasso selects more variables than Lasso, similar to Elastic Net. This could be due to the fact that strong multicollinearity exists in eye data, where the mean absolute pairwise correlation among the 200

variables is 0.606. Group selection property of STRANDS leads to larger model, with more accurately estimated coefficients. STRANDS-Adaptive Lasso also significantly improves the prediction performance over Adaptive Lasso, with larger model.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
Model Size	26	31	3	18	33	10
Prediction Error ($\times 10^{-3}$)	9.23 (0.63)	9.01 (0.60)	14.08 (1.66)	12.05 (1.07)	8.76 (0.48)	9.59 (0.72)
Time (s)	<1	<1	<1	221.55	67.31	121.34

Table 4.14: Real data analysis : eye data, $n = 120, p = 200$

4.5.3 Plasma proteome analysis

Finally we compare STRANDS and its competitors in an analysis of plasma proteome data from Kirk et al. [2011]. Human T lymphotropic virus Type 1 (HTLV-1) is one of the main causes of certain chronic inflammatory diseases, which are commonly referred to as HTLV-1-associated myelopathy/tropic spastic paraparesis (HAM). It is also responsible for adult T cell leukaemia/lymphoma. The study uses surface-enhanced laser desorption mass spectrometry technique to analyse the plasma proteome of 65 HTLV-1-infected patients, who either have HAM disease or are asymptomatic HTLV-1 carriers (AC). The dataset contains log intensities of 49 protein peaks (represented by m/z value in kDa), and the responses are log proviral load and disease status of patients (either HAM or AC). Kirk et al. [2011] and Kirk et al. [2013] used binary disease status as the response and built logistic regression model to find the peaks that most significantly contribute to the discrimination of disease outcomes. We use log proviral load as the response and build linear regression model to find the associations between log proviral load and protein peaks. Since proviral load and disease outcome are closely related, our analysis can be regarded as a continuous relaxation of the binary problem.

The results of plasma proteome analysis are summarised in Table 4.15. Prediction errors are similar across all methods. Adaptive Lasso has the worst predictive performance, and Random Lasso and STRANDS-Adaptive Lasso are marginally

the best. Random Lasso has the largest model size, followed by Elastic Net and Lasso. STRANDS again improve over its base learners, and the improvement is marginal for Lasso, but significant for adaptive Lasso. STRANDS also provides sparser results than its base learners in this example.

For selection, STRANDS-Lasso selects the following peaks: 6.83, 10.8, 11.7, 12.2, 15.7, 17.2, 17.5 and 25.4 kDa; Lasso selects one more peak, 59.5 kDa. We find that peaks selected by linear regression and logistic regression (see below) coincide to some degree. Kirk et al. [2013] apply seven different selection strategies using sparse logistic regression models, and selected protein peaks 10.8, 11.7, 11.9, 13.3, 14.6 17.3, 17.5 and 25.1 kDa. Kirk et al. [2011] applied univariate analysis and logistic regression with Lasso penalty, and selected peaks 11.7, 11.9, and 13.3 kDa. In both studies the authors claimed that 11.7 and 13.3 kDa peaks have the strongest signal, and contribute significantly to predicting patient disease status. Both liner regression and logistic regression select 10.8, 11.7, and 17.5 peaks, and the new peaks selected by linear regression are also interesting and could be investigated further.

	Lasso	ENet	AdaLasso	RLasso	STRD-Lasso	STRD-AdaLasso
Model Size	9	12	6	15	8	5
Prediction Error	0.573 (0.028)	0.568 (0.028)	0.634 (0.033)	0.557 (0.03)	0.566 (0.032)	0.554 (0.028)
Time (s)	<1	<1	<1	264.14	69.19	101.45

Table 4.15: Real data analysis : plasma proteome data, $n = 65, p = 49$

4.6 Conclusions and discussion

In this section we proposed a new variable selection strategy structural randomised selection (STRANDS), in line with Random Lasso. The method inherits some properties from Random Lasso and alleviates some issues of l_1 penalised regression. Particularly, both STRANDS and Random Lasso encourage strongly correlated variables to be selected simultaneously, and the model size is no longer limited by the sample size. STRANDS takes into account the correlation structure to more effectively explore the model space. Compared to Random Lasso, STRANDS'

implementation is more automatic such that less parameters need to be tuned, so the method is computationally less intensive. STRANDS is also easy to be parallelised. Without bootstrap step, STRANDS uses full sample information in all candidate models. Both simulation study and real data study compared the performance of STRANDS with popular regularised regression methods. We found that STRANDS typically improves the performance of its base learner in low-to-moderate dimensional settings, especially when strong multicollinearity exists; in high-dimensional settings the benefit of STRANDS is less clear. Although in this chapter the baseline learners used with STRANDS were Lasso and Adaptive Lasso, STRANDS can also be combined with other sparse regression methods to improve their performance.

Park et al. [2015] proposed recursive random Lasso to improve Random Lasso's computational efficiency and variable selection performance. Specifically, instead of repeatedly applying Lasso to obtain importance measurements, recursive random Lasso calculates importance measures recursively in the modelling process, such that importance measures are recalculated at each iteration using the regression results from the previous iterations, and in each iteration Lasso is applied after variable sampling based on the current important measures. The authors also proposed a parametric test for variable selection. They show that recursive random Lasso has similar selection and prediction performance to Random Lasso in high-dimensional settings, with much more efficient computation. It would be interesting to explore if using recursive approach in STRANDS improves computational efficiency while also retaining STRANDS' good performance.

Both Random Lasso and STRANDS can be interpreted as randomised screening methods. As a pioneering work, Fan and Lv [2008] proposed to use univariate regression coefficients to reduce the dimension of data from ultra-high to manageable in high-dimension linear regression. Specifically, their sure independence screening (SIS) approach first screens the data to have only a number of variables with largest univariate regression coefficients, and subsequent analysis is then performed on the sub-data. The method is very straightforward and is proved to have sure screening property in an asymptotic framework, i.e., the probability of screened sub-data containing all relevant variables converges to one as the sample size approaches infinity. They also propose an iterative sure independence screen-

ing (ISIS) method to enhance the finite sample performance. However, there are two critical issues with this method. Firstly, the univariate regression coefficient can be highly unstable to represent the actual importance of a variable, especially in high-dimensional data with complicated correlation structure. Secondly, there is no justification on how to choose the degree of screening, i.e., how many variables should be retained in the sub-data. In Random Lasso, instead of univariate regression, the importance measure is calculated by l_1 penalisation method on random combination of variables, which captures the multivariate effects and sparsity pattern of data; instead of arbitrary screening degree, Random Lasso tunes the number of screened variables in a data-driven way, although the procedure can be computationally intensive. STRANDS addresses these two issues in an adaptive and data-driven way. As in Random Lasso, we explore variable importance by repeatedly applying penalised regression on random subsets of variables in Step 1, to take into account the multivariate effects instead of univariate effects as in SIS. Multivariate effects are more reliable when dimensionality is high and correlation structure is complex. The degree of screening is automatically obtained based on selection probabilities from Step 1, which is computationally efficient and more accurate.

One critical problem associated with screening-related approaches is screening bias, i.e., if screening and subsequent variable selection are based on the same data, irrelevant variables surviving the screening become overly promising. This problem has been pointed out by several authors, see e.g. Fan and Lv [2008] and discussion therein. Zhu and Yang [2014] prove that naive variable selection on variables surviving screening procedure is problematic, and can lead to significant bias on both variable selection and prediction. In other words, the surviving variables after screening step should not be treated as if they are given in the first place. Instead, they suggest to avoid screening bias by data-splitting, where data is split into two halves, and variable screening is performed on one half, while variable selection on the surviving variables is performed on the other half. They claim that data splitting can significantly reduce the screening bias in variable selection and prediction, although selection consistency is not guaranteed for fixed design. It is because the fact that consistency conditions hold for the full data does not imply that the same conditions hold for a subset of samples. Even if certain data-

splitting can satisfy the required conditions, finding the correct split is difficult in practice.

We address the screening bias in a different way. Unlike Random Lasso, we obtain a model size estimate $\tilde{s}$ from Step 1, which gives guidance on sparsity of the underlying model. Sampling in Step 2 is thus performed in an informed way, with the goal that only relevant variables are likely to survive the screening, and false positives are controlled by $\tilde{s}$. Sampling procedures in Step 2 of STRANDS and Random Lasso acknowledge the fact that estimation in Step 1 is subject to uncertainty of data, so the importance measures of variables should be respected in a stochastic way. This further mitigates the screening bias, and is in contrast to SIS where the ranking of variable importance is exactly followed.

Both data-splitting and bootstrap fail to take advantage of full sample information. Although from results of systematic comparison study in Chapter 2 we observe that subsampling of samples can potentially improve method performance when n is large, the loss of information can cause loss of power and prediction accuracy when n is small relative to s_0 and p, which is common in biomedical studies. Specifically, recall that consistent variable selection requires

$$s_0 log(p) = o(n), \tag{4.2}$$

while data-splitting and bootstrapping essentially reduce the active sample size, which may break the consistency condition.

Although the intuitive values of parameters for STRANDS lead to satisfactory results, alternative choices of these values and implementations can also be of interest in practice. For learning correlation structure, we apply an iterative procedure with a correlation threshold (ρ_0) 0.5. A hierarchical clustering procedure based on correlation can be used instead, similar as in Bühlmann et al. [2013]; this can potentially improve the efficiency of STRANDS in ultra-high dimensional data, although the number of clusters can be difficult to choose, and the interpretation of a correlated group may be different. The correlation threshold should adapt to specific data types and research interest. The general guidance for choosing the correlation threshold is to appropriately capture the correlation structure of data, without introducing too many correlation groups (relative to dimension of data);

otherwise the method could be less stable. $s = \lceil \sum_t \theta_t \rceil$ in Step 2a (i) of STRANDS is an automatic choice for selecting important variables, but finer tuning of s can also be desirable, e.g., based on external knowledge of data, or appropriate thresholding on θ_t's to eliminate the influences of likely irrelevant variables. When sample size is large enough, splitting the data for screening and variable selection separately can further reduce screening bias, without losing much power. Finally, depending on the goal of a specific study, the selection probability threshold π_{thr} can be either more restrictive to better control false positives, or more relaxed to increase the chance of capturing signals.

Chapter 5

Adaptive Ridge forward selection

In this chapter we propose another ensemble learning method, which combines Ridge Regression and forward selection from a Bayesian perspective, aiming to improve the prediction performance of Ridge Regression in sparse settings.

5.1 Introduction

From the systematic comparison study in Chapter 2, we observe that Ridge Regression is typically unsatisfactory in sparse settings since it does not perform variable selection and so too many irrelevant variables are included, negatively impacting the prediction accuracy. It can therefore be desirable to discover the underlying sparsity pattern before applying Ridge Regression. Here, we propose an approach combining Bayesian Ridge Regression (see Section 1.5.4) with a probabilistic forward selection procedure that aims to improve prediction performance of Bayesian Ridge Regression in sparse settings, while keeping its simple closed-form computation.

Forward selection (see Section 1.3.2) is a simple variable selection method that can be used to obtain a sparse model prior to applying Ridge Regression. The procedure starts with the null model, and at each step, the most relevant candidate variable is tested based on certain criteria, and is added to the model if a statistically significant improvement is achieved for the model fit. The procedure stops when no variable can be included according to the criteria. One criterion for model

fit is BIC (1.5); that is, at each stage of forward selection, the candidate variable is included in the model only if the BIC score decreases. In a Bayesian framework, a criterion based on the Bayes factor can be used. The Bayes factor (see Section 1.5.2) of two competing models M_0 and M_1 is $BF_{01} = \frac{P(D|M_0)}{P(D|M_1)}$, which is the ratio of the marginal likelihood, evaluating relative model evidence. $BF_{01} > 1$ indicates that M_0 is more supported by the data than M_1.

Forward selection can be unstable in the sense that slight change in data can lead to large changes in coefficient estimates [Breiman, 1996], especially when variables are correlated. At certain stages of forward selection, the BIC scores for a pair of nested models can be similar or Bayes factor can be close to one, but a deterministic decision is made on model choice; a small change of data can lead to opposite conclusion. In other words, traditional forward selection ignores the uncertainty at each stage, and the corresponding errors accumulate along the forward selection procedure. With this instability, it is difficult to find the best model, and prediction accuracy can be severely impaired.

Bayes factor can also be converted to probability to quantify the relative plausibility of candidate models. Let M_0 and M_1 be two different models, D the observed data, and assign prior probabilities of the two models to be $P(M_0) = p_0$ and $P(M_1) = 1 - p_0$, then the posterior probability of M_0 is given by

$$P(M_0|D) = \frac{BF_{01}}{BF_{01} + (1 - p_0)/p_0}, \tag{5.1}$$

where $BF_{01} = \frac{P(D|M_0)}{P(D|M_1)}$ is the Bayes factor of M_0 versus M_1 [Casella et al., 2009].

We propose a probabilistic forward selection procedure based on Bayes factors for Bayesian Ridge Regression. The procedure produces sparse models, which means that instead of applying Bayesian Ridge Regression to all p variables, it is applied to the selected variables. Importantly, the probabilistic procedure uses posterior model probabilities (5.1) to take into account uncertainty in forward selection as opposed to using hard-threshold decisions (based on e.g. BIC or Bayes factors) that are deterministic. We average over an ensemble of models produced by probabilistic forward selection, which is necessary to stabilise the coefficient estimates, and consequently improve the prediction accuracy.

The remainder of the section is organised as follows. In Section 5.2 we describe

and motivate the proposed method. In Section 5.3 we show results comparing the proposed method with Lasso and Bayesian Ridge Regression using simulated datasets from Section 4.3.1. We conclude with a discussion in Section 5.4.

5.2 Method

Recall from Section 1.5.4 that the Bayesian Ridge Regression can be presented as follows:

$$\begin{aligned} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\beta}, \sigma^2) &= N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}) \\ \boldsymbol{\beta}|\sigma^2, \lambda &\sim N\Big(\mathbf{0}, \frac{\sigma^2}{\lambda}\mathbf{I}\Big) \\ \sigma^2 &\sim IG(a, b), \end{aligned} \tag{5.2}$$

where $\lambda > 0$ controls the degree of shrinkage. Point estimates have closed form solution $\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$, and the marginal distribution of $\mathbf{Y}$ follows a multivariate Student t-distribution,

$$p(\mathbf{Y}|\mathbf{X}, \lambda) = \int P(\mathbf{Y}|\boldsymbol{\beta}, \sigma^2)P(\boldsymbol{\beta}|\sigma^2)P(\sigma^2)d\boldsymbol{\beta}d\sigma^2 = MVSt_{2a}(\mathbf{0}, \frac{b}{a}(\mathbf{I} + \frac{\mathbf{X}\mathbf{X}^T}{\lambda})). \tag{5.3}$$

We specify the prior for σ^2 to be non-informative, i.e., a, b are small values; we take $a = b = 0.001$. We fix λ to be $1/n$, which is the expectation of the " default prior" motivated by the Zellner-Siow prior [Liang et al., 2008], i.e., $\mathcal{G}(\frac{1}{2}, \frac{n}{n})$ and $\mathcal{G}(.,.)$ denotes the gamma distribution. Note that $1/n$ is also the fixed unit information prior [Kass and Raftery, 1995].

For a pair of nested models M_0 and M_1, Bayes factor BF_{01} is calculated as

$$BF_{01} = \frac{p(\mathbf{Y}|\mathbf{X}, M_0, \lambda)}{p(\mathbf{Y}|\mathbf{X}, M_1, \lambda)}, \tag{5.4}$$

and $p(\mathbf{Y}|\mathbf{X}, M_i, \lambda)$ is the marginal likelihood of $\mathbf{Y}$ given model M_i. Our method combines the idea of forward selection and Bayes factor for Ridge Regression to obtain sparse solution. At each step of forward selection we compare a pair of nested models M_0 and M_1, where M_1 has an additional variable compared to

M_0. We use the posterior model probabilities in (5.1) to make this comparison. It is natural choice to assign an equal prior probability to the two models, i.e., $P(M_0) = P(M_1) = 0.5$. Then (5.1) becomes

$$P(M_0|\mathbf{X}, \mathbf{Y}) = \frac{BF_{01}}{BF_{01} + 1}, \tag{5.5}$$

and $P(M_1|\mathbf{X}, \mathbf{Y}) = \frac{1}{BF_{01}+1}$. In other words, we include the additional variable in M_1 with probability $P(M_1|\mathbf{X}, \mathbf{Y})$. After completion of the probabilistic forward selection procedure, we have a selected set of variables. We apply Bayesian Ridge Regression to this selected set to obtain coefficient estimates. Probabilistic forward selection and coefficient estimation are repeated to obtain an ensemble of candidate models, and final coefficients are averaged across candidate models that pass a quality control step. This step eliminates models that have marginal likelihood smaller than the null model. Each individual candidate model may miss certain signals due to stochasticity, but averaging across the ensemble will provide accurate coefficient estimates.

Forward selection requires an initial ranking of variables to determine the order they are considered for inclusion in the model, and we simply rank the variables using absolute coefficient estimates from applying Bayesian Ridge Regression to all variables.

Pseudocode for the proposed adaptive Ridge forward selection procedure is provided in Algorithm 3, and the forward selection procedure is illustrated in Figure 5.1.

5.3 Results

In this section we compare the adaptive Ridge forward selection (referred to as RFS hereafter) with Lasso, STRD-Lasso and standard Bayesian Ridge Regression using three datasets from previous analyses. We compare the methods using Example 4.3, 4.4 and 4.5 from Section 4.3. All three examples are sparse settings, and specifically, in Example 4.3 relevant variables are strongly correlated with coefficients of different signs; in Example 4.4 variables are orthogonal to each other; in Example 4.5 relevant variables are in several moderately correlated blocks. We

Algorithm 3 Adaptive Ridge forward selection (RFS)

Input: n by p design matrix $\mathbf{X} = (\mathbf{x_1} \dots \mathbf{x_p})$ and response vector $\mathbf{Y}$; number of iterations B

Output: coefficient estimates $\hat{\beta}_j$ for each variable

Step 1:
Run Bayesian Ridge Regression on $(\mathbf{Y}, \mathbf{X})$ with $\lambda = 1/n$, and obtain coefficient estimate $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\boldsymbol{X}^T\mathbf{Y}$. Rank variables by their absolute coefficients, from largest to smallest. Denote the ranked variables as $\boldsymbol{x^{(1)}, \dots, x^{(p)}}$, where coefficient of $\boldsymbol{x^{(k)}}$ is not smaller than that of $\boldsymbol{x^{(k+1)}}$, $k = 1 \dots p-1$.

Step 2:
for $b = 1, \dots, B$ **do**
  $\mathbf{M} = \emptyset$
  for $j = 1, \dots, p$ **do**
    1. Let current set of selected variables be $\boldsymbol{M_0} = \mathbf{M}$, and $\boldsymbol{M_1} = \boldsymbol{M_0} \cup \boldsymbol{x^{(j)}}$
    2. Calculate Bayes factor $BF_{01} = \frac{p(\mathbf{Y}|\mathbf{X}, \boldsymbol{M_0}, \lambda)}{p(\mathbf{Y}|\mathbf{X}, \boldsymbol{M_1}, \lambda)}$ of $\boldsymbol{M_0}$ and $\boldsymbol{M_1}$
    3. Calculate the posterior probability of model $\boldsymbol{M_1}$ as $\mathrm{P}(\boldsymbol{M_1}|\mathbf{X}, \mathbf{Y}) = \frac{1}{BF_{01}+1}$. This is also the selection probability of $\boldsymbol{x^{(j)}}$
    4. Draw a random number z from $\mathrm{Bern}(\mathrm{P}(\boldsymbol{M_1}|\mathbf{X}, \mathbf{Y}))$, if $z = 1$, $\boldsymbol{M} = \boldsymbol{M_1}$; otherwise $\boldsymbol{M} = \boldsymbol{M_0}$
  end for

  Apply Bayesian Ridge Regression on $(\boldsymbol{M}, \mathbf{Y})$ with $\lambda = 1/n$, and obtain estimates $\hat{\beta}_j^{(b)}$ for those $\boldsymbol{x_j} \in \mathbf{M}$, and $\hat{\beta}_j^{(b)} = 0$ if $\boldsymbol{x_j} \notin \mathbf{M}$.
end for

Step 3:
Remove those iterations where the final selected model has smaller marginal likelihood than the null model
Denote the rest qualified iterations as $\mathcal{I}$, then the final coefficient estimates are $\hat{\beta}_j = \frac{\sum_{b \in \mathcal{I}} \hat{\beta}_j^{(b)}}{|\mathcal{I}|}$, $j = 1 \dots p$.

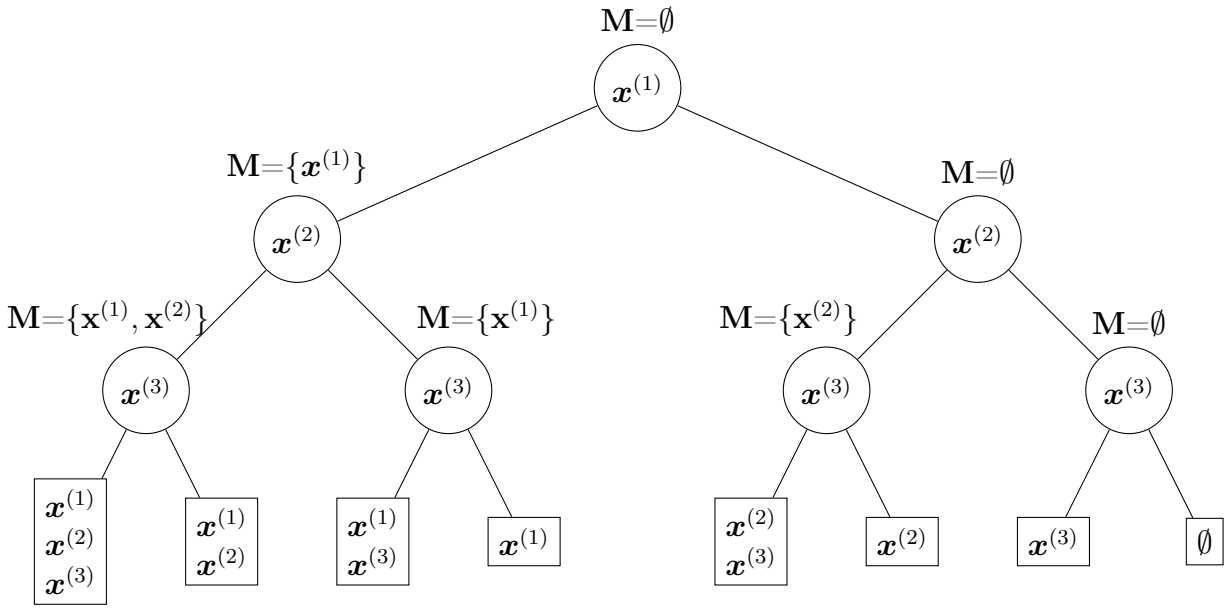


Figure 5.1: An illustration of adaptive Ridge forward selection with three variables. Variables are ranked by their absolute coefficient estimates, and are tested following the order of $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ and $\mathbf{x}^{(3)}$. Model is initialised to be $\emptyset$ and includes selected variables along forward selection procedure. Each internal node represents a decision on selection of the corresponding variable based on probability. The left branch represents inclusion of the variable while the right branch represents exclusion. Leaf nodes represent possible resulting models at the end of the forward selection procedure.

use $B = 300$ for RFS and show results for $n = 50$ and $n = 100$. We focus on comparison of prediction accuracy, and the results are summarised in Tables 5.1, 5.2 and 5.3. We see in all three scenarios RFS has a significant improvement on prediction over standard Bayesian Ridge Regression, especially in high-dimensional case where only a small proportion of variables are relevant. RFS remains competitive compared to Lasso, especially when strong multicollinearity exists as in Table 5.1; RFS's predictive performance is similar to that of STRD-Lasso in most settings. The proposed approach is tuning-free, and it is also computationally efficient, since closed-form solution is available at each stage of forward selection. Specifically, computation time of RFS for $n = 50, 100$ are 3.75s and 4.05s for Example 4.3, 61.22s and 70.41s for Example 4.4, 78.14s and 81.79s for Example 4.5.

RFS does not perform variable selection *per se* in the sense that most coefficient estimates tend to be non-zero. However, one could attempt to discriminate relevant variables from irrelevant ones based on the magnitudes of coefficient estimates obtained using RFS. Figure 5.2 shows the distribution of coefficient estimates of RFS in Example 4.3. We see that there is a clear discrimination of coefficient estimates between relevant and irrelevant variables. A selection rule could be developed based on such discrimination.

MSE	Lasso	RFS	Ridge	STRD-Lasso
$n = 50$	6.4 (0.185)	4.85 (0.14)	8.48 (0.15)	5.13 (0.21)
$n = 100$	5.08 (0.11)	3.73 (0.09)	6.75 (0.08)	2.32 (0.11)

Table 5.1: Example 4.3: $p = 40, p_0 = 10, \sigma = 3$. One block of 10 pairwise correlated variables (correlation $\rho = 0.9$), all of which are signals, with coefficients 3, 3, 3, 3, 3, -2, -2, -2, -2, -2. The other 30 variables are independent from each other, and independent from the 10 correlated variables, all of which are noise variables. Results are averages across 100 replicates, with bootstrap standard errors in parentheses.

5.4 Discussion

In this section we proposed a novel probabilistic forward selection method based on Bayesian Ridge Regression. The method takes into account the uncertainty at

MSE	Lasso	RFS	Ridge	STRD-Lasso
$n = 50$	61.9 (3.25)	62.35 (2.21)	105.89 (0.37)	62.11 (2.92)
$n = 100$	8.33 (0.37)	7.32 (0.62)	87.85 (0.45)	7.36 (0.32)

Table 5.2: Example 4.4: $p = 300, p_0 = 10, \sigma = 3$. Variables are i.i.d. drawn from $N(\mathbf{0}, \mathbf{I})$. 10 signals are randomly allocated among p variables, with coefficients 3, 3, 3, 3, 3, 4, 4, 4, 4, 4. Results are averages across 100 replicates, with bootstrap standard errors in parentheses.

MSE	Lasso	RFS	Ridge	STRD-Lasso
$n = 50$	59.65 (2.15)	53.44 (2.0)	57.81 (0.69)	51.34 (1.69)
$n = 100$	9.71 (0.43)	10.39 (0.66)	49.40 (0.32)	9.83 (0.41)

Table 5.3: Example 4.5: $p = 300, p_0 = 10, \sigma = 3$, 10 blocks of 10 pairwise correlated variables ($\rho = 0.7$), each having one signal. Non-zero coefficients are 3, 3, 3, 3, 3, 4, 4, 4, 4, 4. Results are averages across 100 replicates, with bootstrap standard errors in parentheses.

each stage of forward selection, and uses Bayes factor in the form of probability to make decisions on model selection in a stochastic way.

The stochastic forward selection procedure can be represented by a rooted binary tree (see Figure 5.1). Each internal node represents a probabilistic test on whether a variable should be retained in the model, based on posterior model probability. Compared to a deterministic decision based on criteria such as BIC, this probabilistic test takes uncertainty into account at each split. Two branches from the node represent two possible decisions of the test. The nodes are ordered by variables' importance measures defined by Ridge Regression estimates, such that important variables are tested before less important ones. Each path from the root node to a leaf node corresponds to a candidate sparse model. We average coefficient estimates across multiple candidate sparse models to reduce variance.

In a similar spirit as STRANDS, RFS first calculates absolute coefficients as importance measures of each variable, and variable selection is then guided by those importance measures. The ranking of variables is important since the earlier a variable enters the model, the better chance it will be retained. In other words, RFS implicitly puts larger weights on variables with larger importance measures. As seen from previous results, Ridge Regression can have inferior ranking perfor-

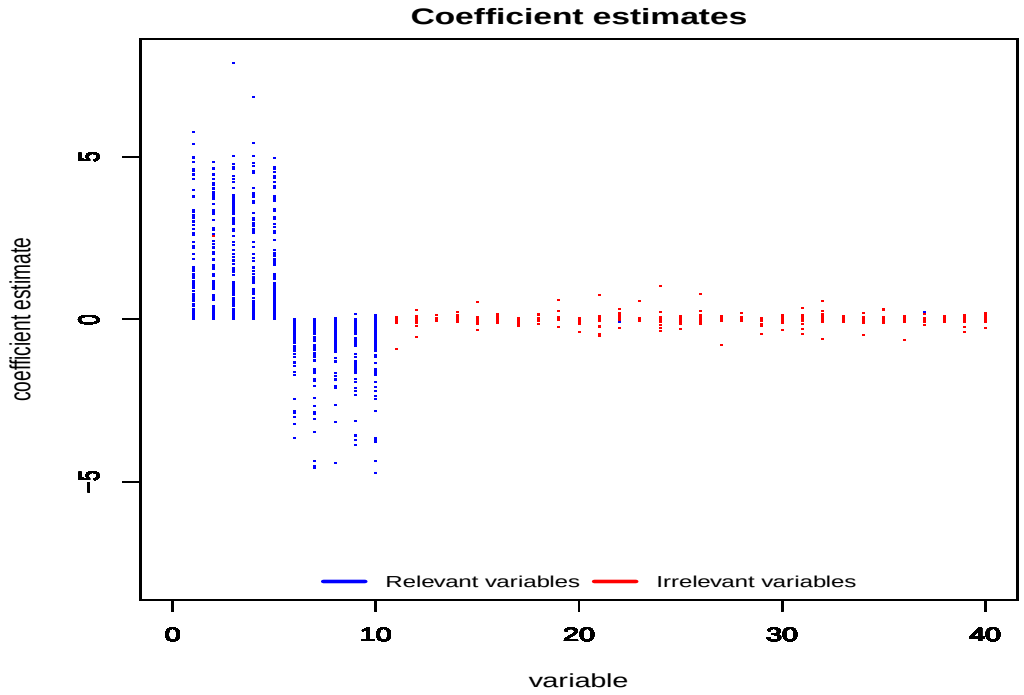


Figure 5.2: Coefficient estimates of RFS in Example 4.3, $n = 100, p = 40$. Simulation is repeated 100 times, and each dot corresponds to a coefficient estimate of a variable (represented by its location on x axis) in one simulation. Relevant variables and irrelevant variables are represented by blue and red dots respectively.

mance in high-dimensional settings, but since ranking is respected in a stochastic way rather than deterministic, the ranking issue is alleviated. In each resulting model after forward selection procedure, relevant variables are likely to be selected, and with reduced dimensionality, prediction tends to be more accurate.

The prior model probabilities on candidate models at each stage of forward selection is essential. In general, we need to assign prior distribution to the whole model space containing 2^p candidate models, and the choice of such prior can be tricky. An standard approach is to make prior assumptions on indicator variables $\boldsymbol{\gamma} = (\gamma_1 \dots \gamma_p)$, defined as $\gamma_j = 1$ if $\beta_j \neq 0$ and $\gamma_j = 0$ if $\beta_j = 0$. Such specification can lead to Beta-Binomial prior on the model space. Specifically, in linear model

(1.1), let $P(\gamma_j = 1) = \theta$, $j = 1 \dots p$, then

$$\begin{aligned} p(\boldsymbol{\gamma}|\theta) &= \theta^{|\boldsymbol{\gamma}|}(1-\theta)^{p-|\boldsymbol{\gamma}|} \\ \theta &\sim Beta(a_\theta, b_\theta). \end{aligned} \tag{5.6}$$

is called the Beta-Binomial prior. This prior does not penalise model complexity appropriately if a_θ and b_θ are fixed, and if a_θ and b_θ are calculated using data, more justifications are necessary and such estimate can also be inaccurate [Womack et al., 2015]. In our case, since the comparison at each stage of forward selection is restricted to a pair of nested models with only one variable different, the choice of equal prior probabilities on candidate models seems reasonable.

Instead of sampling from the posterior distribution over the model space, adaptive Ridge forward selection generates a quasi-posterior sample of models as an ensemble, using a stochastic greedy search. Our approach has some similarities to the full Bayesian approach, as they both put higher weights on important variables, and models with less credibility are down-weighted. However, our approach does not result in a true posterior, and the advantage is computational efficiency since it is based on a greedy search.

Leday et al. [2017] propose a similar approach in gene network reconstruction context. They first rank variables by Ridge Regression coefficient estimates, and perform forward selection in a similar way. However there are two key differences compared to our approach. Firstly, they calculate p_0 from data instead of fixing $p_0 = 0.5$. Secondly, their prime focus is variable selection and false positive control. At each stage of forward selection, they set a (pre-specified) threshold on posterior null probability $P(M_0|\mathbf{X}, \mathbf{Y})$, such that the variable selection is a deterministic procedure. In sparse settings, estimated p_0 can be small, such that we may sacrifice statistical power for false positive control, which may not be optimal for our prediction purpose. There is a large amount of uncertainty associated with forward selection, especially in high-dimensional settings, and we deal with the uncertainty via ensemble learning approach, which can significantly reduce the variance in estimation.

Chapter 6

Conclusions and future work

6.1 Summary

Recent years have witnessed the arise of high-dimensional data in various areas. For example, computer vision uses images to recognise patterns, and each image contains a huge number of pixels, which can be viewed as variables. Financial data can also be high-dimensional, especially for time series data. Functional magnetic resonance imaging (fMRI) data has many more voxels than the number of subjects, and voxels can be viewed as variables. High-dimensional data is common in modern biological science, since high-throughput technology enables us to measure a huge number of features on the molecular level.

Traditional statistical approaches are no longer appropriate for high-dimensional data. High dimensionality and salient characteristics associated with high-dimensional data in practice greatly complicate the regression problem. To name a few, spurious correlations between relevant and irrelevant variables are more likely to exist by chance, so is incidental endogeneity where variables correlate with the error terms [Fan et al., 2014]; statistical algorithms tend to be less stable when the search space is huge, and computation can be demanding. In addition, high noise level, insufficient sample size and mixture of unknown populations are also common in certain areas. New challenges have emerged, requiring new statistical thinking and methodology. Regularisation or dimension reduction is required to get sensible results, and in this thesis we focussed on penalised regression methods.

In Chapter 2 we compared popular penalised regression methods in a system-

atic way, varying different features of data, and in Chapter 3 we extended the comparison to different model assumptions and parameter specifications. Finally in Chapter 4 and Chapter 5 we proposed two ensemble learning methods, STructural RANDomised Selection (STRANDS) to improve sparse penalised regression methods, and adaptive Ridge forward selection to improve the prediction performance of Ridge Regression.

6.2 Conclusions

In Chapter 2, we systematically compared different state-of-the-art penalised regression methods in the context of high-dimensional data, in a large number of data-generating scenarios, with varied characteristics. Many factors can be responsible for accuracy of results in linear regression, and in this thesis we mainly focused on the influence of $r = n/(s_0 \log(p - s_0))$, SNR and correlation structure. Although these three factors are not independent from each other, they capture three different data properties, namely, sufficiency of sample size with respect to dimensionality and sparsity, strengths of signals in contrast to noise level, and degree of multicollinearity. While each characteristic is not decisive when considered in isolation, their joint information largely determines the accuracy of the model. We found that the relative performance between methods depended on all these factors, and also on the goal of the study. No method was unambiguously dominant across the majority of scenarios. However, we were able to provide insight on when certain methods should be preferred, which can be useful in practice. In a nutshell, Lasso was reliable in most scenarios, and increasing l_2 penalty was beneficial only in extreme correlation designs; Dantzig Selector performed similarly to Lasso, but was more computationally intensive; Ridge Regression was found to perform badly in most settings; SCAD and Stability Selection were preferred in the "easy" settings, but could be too conservative when settings became more difficult.

Although in our simulation study the underlying model was known in advance, which is typically not true in practice, one can observe and have reasonable guess at characteristics of the underlying model, based on domain knowledge of the data. For example, when analysing some polygenic traits, one may expect more than a

few relevant variables, i.e., large s_0; GWAS study is typically known to be noisy, i.e., one should expect low SNR. With these information, one can refer to our conclusions to choose the best method for their research purpose. In Chapter 3 we extended the simulation study to evaluate effects of parameter specification and model assumptions on relative performance between methods. We found that in practice where model assumptions can be violated and data can have diverse characteristics, choosing the best method or tuning parameters requires thorough consideration. Our results and code can also serve as a resource for users to compare other penalised regression methods or benchmark new methodology in scenarios with various characteristics. However, we acknowledge that some data characteristics are difficult to be extracted without external information, such as how many correlated variables are actually relevant. This is a limitation of our study.

In the comparison study we found that correlation can severely affect method performance, and Ridge Regression typically has poor prediction accuracy in sparse settings. These aspects motivated us to develop two randomised approaches to mitigate the defects of penalised regression methods that were brought to light.

In Chapter 4 we proposed STructural RANDomised Selection (STRANDS) to improve the performance of sparse penalised regression methods, particularly in data with strong correlation. It can be viewed as a randomised screening method. It uses multivariate effects to determine the variable importance and degree of screening, and dimension reduction is achieved in a stochastic way to acknowledge the uncertainty associated with the estimated quantities. Sampling of variables encourages correlated variables to be selected simultaneously, and the estimated degree of screening ensures that the right amount of variables are involved in each candidate model, such that screening bias is well controlled. Through simulated and real data, we showed that the proposed method typically improves upon its base learners in low-to-moderate-dimensional settings, especially for data with strong multicollinearity. Compared to Random Lasso, our method has better bias control.

In Chapter 5 we proposed adaptive forward Ridge selection which combines Bayesian Ridge Regression and forward selection. The approach aims to take into account the large amount of uncertainty in forward selection by converting Bayes

factors into probabilities, which guide forward selection in a stochastic way. Our results showed that having a sparse set of variables can significantly improve the prediction performance of Ridge Regression.

6.3 Future directions

In the comparison study in Chapter 2, more penalised linear regression methods could be covered to reveal their practical properties. A comparison between frequentist and Bayesian methods would also be desirable, although it might be challenging to make the comparison fair, since Bayesian methods typically have a number of hyper-parameters to set, which could have a large impact on method performance. The scope of the comparison could also be extended to see how relative performance is affected when certain assumptions for the linear model are violated. For example, in Section 3.5 effect of non-Gaussian noise was investigated. Throughout the thesis we only focused on linear regression, and it would be interesting to investigate how our conclusions may extend to the non-linear regression setting. While we expect the conclusions to be largely consistent with those of linear regression setting, non-linear regression settings have different model assumptions and data properties, which require careful consideration. Candes et al. [2018] is an example of recent work on variable selection in high-dimensional non-linear models. The authors proposed a general framework to achieve controlled variable selection (i.e., control the false positive rate), which guarantees valid inference for finite samples, with arbitrary conditional distribution of the response.

In Step 1 of structural randomised selection and Random Lasso, results across different models are directly averaged, where these models can be very different from each other. A more informed way of averaging would be to assign different weights to results of different models, such that results of more informative models are given higher weights. In this way, the ensemble learning method would be more robust against unfortunate selection of poor models by chance.

In adaptive forward Ridge selection, we assign an equal prior belief to any pair of nested models in the forward selection procedure, and sensitivity analysis is useful to see how much results will change with perturbation of the prior belief. Our proposed approach does not result in a fixed selected set of variables. In order

to achieve sparse representation, one needs to come up with a sensible thresholding rule on the coefficient estimates.

6.4 Final remark

Data in practice typically has complex characteristics, and a good regression method should be robust against idiosyncrasies in the data generation process and be able to efficiently explore the model space. In this thesis, we have shown the importance of comprehensive empirical evaluation of methods. It would be beneficial to see more of these studies performed in the future to better understand finite-sample properties of existing methods. We have also found that using methods within ensemble learning procedures can be helpful to take uncertainty into account and stabilise method performance. Since ensemble learning can often result in improved model quality, it will continue to play an important part in method development and the analysis of complex data.

References

Bach, F. R. (2008). Bolasso: Model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 33–40, New York, NY, USA. ACM. 13, 25

Bickel, P. J., Ritov, Y., and Tsybakov, A. B. (2009). Simultaneous analysis of Lasso and Dantzig selector. *The Annals of Statistics*, 37(4):1705–1732. 18

Bondell, H. D. and Reich, B. J. (2012). Consistent high-dimensional Bayesian variable selection via penalized credible regions. *Journal of the American Statistical Association*, 107(500):1610–1624. 34

Breheny, P. and Huang, J. (2011). Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *The Annals of Applied Statistics*, 5(1):232–253. 37

Breiman, L. (1994). Bagging predictors. Technical Report 421, Department of Statistics, University of California. 25

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140. 24, 25, 127

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32. 27

Brown, G., Wyatt, J., Harris, R., and Yao, X. (2005). Diversity creation methods: A survey and categorisation. *Journal of Information Fusion*, 6:5–20. 24

Bühlmann, P. and Mandozzi, J. (2014). High-dimensional variable screening and bias in subsequent inference, with an empirical comparison. *Computational Statistics*, 29(3):407–430. 34, 36, 62, 65

Bühlmann, P., Rütimann, P., van de Geer, S., and Zhang, C.-H. (2013). Correlated variables in regression: clustering and sparse estimation. *Journal of Statistical Planning and Inference*, 143(11):1835–1858. 98, 124

Bühlmann, P. and van de Geer, S. (2011). *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Publishing Company, Incorporated, 1st edition. 33

Bühlmann, P. and van de Geer, S. (2015). High-dimensional inference in misspecified linear models. *Electronic Journal of Statistics*, 9(1):1449–1473. 94

Candes, E., Fan, Y., Janson, L., and Lv, J. (2018). Panning for gold: 'model-X' knockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society (Series B)*, 80:551–577. 139

Candes, E. and Tao, T. (2007). The Dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, 35(6). 18

Casella, G., Girón, F. J., Martínez, M. L., and Moreno, E. (2009). Consistency of bayesian procedures for variable selection. *The Annals of Statistics*, 37(3):1207–1228. 127

Celeux, G., Anbari, M. E., Marin, J.-M., and Robert, C. P. (2012). Regularization in regression: Comparing bayesian and frequentist methods in a poorly informative situation. *Bayesian Analysis*, 7(2):477–502. 34

Chatterjee, S. (2013). Assumptionless consistency of the Lasso. *ArXiv e-prints*. arXiv:1303.5817. 14

Chen, J. and Chen, Z. (2008). Extended bayesian information criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771. 10

Chiang, A. P. et al. (2006). Homozygosity mapping with SNP arrays identifies TRIM32, an E3 ubiquitin ligase, as a Bardet-Biedl syndrome gene (BBS11). *Proceedings of the National Academy of Sciences of the United States of America*, 103(16):6287–6292. 119

Costa-Silva, J., Domingues, D., and Lopes, F. M. (2017). RNA-Seq differential expression analysis: An extended review and a software tool. *PloS one*, 12(12):e0190152. 1

Denison, D. G. T., Holmes, C. C., Mallick, B. K., and Smith, A. F. M. (2002). *Bayesian Methods for Nonlinear Classification and Regression*. Wiley. 24

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32(2):407–499. 7, 8, 14

Efron, B., Hastie, T., and Tibshirani, R. (2007). Discussion: The Dantzig selector: Statistical estimation when p is much larger than n. 35(6):2358–2364. 43

Fan, J., Han, F., and Liu, H. (2014). Challenges of big data analysis. *National Science Review*, 1:293–324. 136

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360. 16, 17, 36

Fan, J. and Lv, J. (2008). Sure independence screening for ultrahigh dimensional feature space (with discussion). *Journal of the Royal Statistical Society Series B*, 70(5):849–911. 97, 122, 123

Fan, J. and Lv, J. (2010). A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 20(1):101–148. 17

Fan, J., Peng, H., et al. (2004). Nonconcave penalized likelihood with a diverging number of parameters. *The Annals of Statistics*, 32(3):928–961. 17

Freund, Y. and Schapire, R. E. (1999). A short introduction to boosting. In *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1401–1406. Morgan Kaufmann. 24

Friedman, J., Hastie, T., and Tibshirani, R. (2010). A note on the group lasso and a sparse group lasso. *ArXiv e-prints*. arXiv:1001.0736. 14, 19

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22. 37

Hastie, T., Tibshirani, R., and Friedman, J. (2008). *The Elements of Statistical Learning (2nd edition)*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA. 28

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67. 12

James, G. M., Radchenko, P., and Lv, J. (2009). DASSO: connections between the Dantzig selector and Lasso. *Journal of the Royal Statistical Society Series B*, 71(1):127–142. 18

Kass, R. E. and Raftery, A. E. (1995). Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795. 21, 128

Kirk, P., Witkover, A., Bangham, C., Richardson, S., Lewin, A., and Stumpf, M. (2013). Balancing the robustness and predictive performance of biomarkers. *Journal of Computational Biology*, 20:979–989. 65, 120, 121

Kirk, P., Witkover, A., Courtney, A., Lewin, A., Wait, R., Stumpf, M., Richardson, S., Taylor, G., and Bangham, C. (2011). Plasma proteome analysis in HTLV-1-associated myelopathy/tropical spastic paraparesis. *Retrovirology*, 8(1):81. 120, 121

Knight, K. and Fu, W. (2000). Asymptotics for lasso-type estimators. *The Annals of Statistics*, 28(5):1356–1378. 18

Krämer, N., Schäfer, J., and Boulesteix, A.-L. (2009). Regularized estimation of large-scale gene association networks using graphical gaussian models. *BMC Bioinformatics*, 10(1):384. 18

Leday, G. G. R., de Gunst, M. C. M., Kpogbezan, G. B., van der Vaart, A. W., van Wieringen, W. N., and van de Wiel, M. A. (2017). Gene network reconstruction using global-local shrinkage priors. *The Annals of Applied Statistics*, 11(1):41–68. 135

Li, X., Zhao, T., Yuan, X., and Liu, H. (2015). The flare package for high dimensional linear regression and precision matrix estimation in R. 16(1):553–557. 37

Liang, F., Paulo, R., Molina, G., Clyde, M. A., and Berger, J. O. (2008). Mixtures of g-priors for bayesian variable selection. *Journal of the American Statistical Association*, pages 410–423. 128

Lim, C. and Yu, B. (2016). Estimation stability with cross-validation (ESCV). *Journal of Computational and Graphical Statistics*, 25:464–492. 65

Lu, T., Pan, Y., Kao, S.-Y., et al. (2004). Gene regulation and DNA damage in the ageing human brain. *Nature*, 429:883–891. 118

Meinshausen, N. and Bühlmann, P. (2006). High-dimensional graphs and variable selection with the Lasso. *Annals of Statistics*, 34(3):1436–1462. 14

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society, Series B*, 72:417–473. 25, 26, 29, 34

Meinshausen, N., Rocha, G., and Yu, B. (2007). Discussion: A tale of three cousins: Lasso, L2 Boosting and Dantzig. *The Annals of Statistics*, 35(6):2373–2384. 18, 43

Morris, J. S. (2012). Statistical methods for proteomic biomarker discovery based on feature extraction or functional modeling approaches. *Statistics and its interface*, 5(1):117–135. 2

Ogutu, J. O., Schulz-Streeck, T., and Piepho, H.-P. (2012). Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions. *BMC Proceedings*, 6(S2):1–6. 12

Park, H., Imoto, S., and Miyano, S. (2015). Recursive Random Lasso (RRLasso) for identifying anti-cancer drug targets. *PLOS ONE*, 10(11):1–19. 122

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. 66

Rakyan, V. K., Down, T. A., Balding, D. J., and Beck, S. (2011). Epigenome-wide association studies for common human diseases. *Nature Reviews Genetics*, 12:529–541. 2

Rudan, I., Campbell, H., Polasek, O., and Prokopenko, I. (2016). *GWAS: The Rise of Hypothesis-Free Biomedical Science : Could Genome-Wide Association Studies (GWAS) Transform Modern Medicine?* Academic Press, Orlando, FL, USA, 1st edition. 1

Scheetz, T. E., Kim, K.-Y. A., Swiderski, R. E., et al. (2006). Regulation of gene expression in the mammalian eye and its relevance to eye disease. *Proceedings of the National Academy of Sciences*, 103(39):14429–14434. 119

Shao, J. and Deng, X. (2012). Estimation in high-dimensional linear models with deterministic design matrices. *The Annals of Statistics*, 40(2):812–831. 12

Sill, M., Hielscher, T., Becker, N., and Zucknick, M. (2014). c060: Extended inference with Lasso and Elastic-Net regularized cox and generalized linear models. *Journal of Statistical Software*, 62(5):1–22. 37

TCGA (2011). Integrated genomic analyses of ovarian carcinoma. *Nature*, 7353(474):609–615. 61

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society*, 58(1):267–288. 13, 30, 105

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society Series B*, pages 91–108. 19

Tibshirani, R. J. and Taylor, J. (2011). The solution path of the generalized lasso. *Annals of Statistics*, 39(3):1335–1371. 15

Tucker, S. L., Gharpure, K., Herbrich, S. M., et al. (2014). Molecular biomarkers of residual disease after surgical debulking of high-grade serous ovarian cancer. *Clinical Cancer Research*, 20(12):3280–3288. 61

Van Der Maaten, L., Postma, E., and Van den Herik, J. (2009). Dimensionality reduction: a comparative review. *Journal of Machine Learning Research*, 10:66–71. 96

Wagenmakers, E.-J. (2007). A practical solution to the pervasive problems of p values. *Psychonomic Bulletin & Review*, 14(5):779–804. 21

Wainwright, M. J. (2009). Sharp thresholds for high-dimensional and noisy sparsity recovery using L1-constrained quadratic programming (Lasso). *IEEE Transactions on Information Theory*, 55(5):2183–2202. 14, 40

Wang, F., Mukherjee, S., Richardson, S., and Hill, S. M. (2018). High-dimensional regression in practice: an empirical study of finite-sample prediction, variable selection and ranking. *ArXiv e-prints*. arXiv:1808.00723. i

Wang, S., Nan, B., Rosset, S., and Zhu, J. (2011). Random lasso. *Annals of Applied Statistics*, 5(1):468–485. 28, 29, 97, 105, 106

Wang, Y. and Zhu, L. (2017). Research and implementation of SVD in machine learning. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, pages 471–475. 96

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241–259. 24

Womack, A. J., Fuentes, C., and Taylor-Rodriguez, D. (2015). Model Space Priors for Objective Sparse Bayesian Regression. *ArXiv e-prints*. arXiv:1511.04745. 135

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67. 19

Zeng, L. and Xie, J. (2012). Group variable selection via SCAD-L2. *Statistics: A Journal of Theoretical and Applied Statistics*, 48(1):49–66. 19

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 38(2):894–942. 19

Zhang, C.-H. and Huang, J. (2008). The sparsity and bias of the Lasso selection in high-dimensional linear regression. *The Annals of Statistics*, 36(4):1567–1594. 94

Zhao, P. and Yu, B. (2006). On model selection consistency of lasso. *Journal of Machine Learning Research*, 7:2541–2563. 13

Zhu, X. and Yang, Y. (2014). Variable selection after screening: with or without data splitting? *Computational Statistics*, 30(1):191–203. 104, 123

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):3133–3164. 17, 18

Zou, H. (2010). Discussion of "Stability selection" by Nicolai Meinshausen and Peter Buhlmann. *Journal of the Royal Statistical Society, Series B*, 72:468. 93

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67(2):301–320. 12, 15, 16

Zuber, V. and Strimmer, K. (2011). High-dimensional regression and variable selection using CAR scores. *Statistical Applications in Genetics and Molecular Biology*, 10(1):1–27. 118